EXTRACTION OF CONTEXTUAL KNOWLEDGE

AND AMBIGUITY HANDLING FOR ONTOLOGY IN VIRTUAL ENVIRONMENT

A Dissertation

by

HYUN SOO LEE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

August 2010

Major Subject: Industrial Engineering

Extraction of Contextual Knowledge and Ambiguity Handling

for Ontology in Virtual Environment

EXTRACTION OF CONTEXTUAL KNOWLEDGE

AND AMBIGUITY HANDLING FOR ONTOLOGY IN VIRTUAL ENVIRONMENT

A Dissertation

by

HYUN SOO LEE

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Amarnath Banerjee
Committee Members,	Lewis Ntaimo
	James A. Wall
	Yoonsuck Choe
Head of Department,	Brett A. Peters

August 2010

Major Subject: Industrial Engineering

ABSTRACT

Extraction of Contextual Knowledge and Ambiguity Handling
for Ontology in Virtual Environment. (August 2010)
Hyun Soo Lee, B.S., Sungkyunkwan University;
M.S., Pohang University of Science and Technology
Chair of Advisory Committee: Dr. Amarnath Banerjee

This dissertation investigates the extraction of knowledge from a known environment. *Virtual ontology* – the extracted knowledge – is defined as a structure of a virtual environment with semantics. While many existing 3D reconstruction approaches can generate virtual environments without structure and related knowledge, the use of Metaearth architecture is proposed as a more descriptive data structure for *virtual ontology*. Its architecture consists of four layers: interactions and relationships between virtual components can be represented in the virtual space layer; and the library layers contribute to the design of large-scale virtual environments with less redundancy; and the mapping layer links the library layer to the virtual space layer; and the ontology layer functions as a context for the extracted knowledge.

The dissertation suggests two construction methodologies. The first method generates a scene structure from a 2D image. Unlike other scene understanding techniques, the suggested method generates scene ontology without prior knowledge and human intervention. As an intermediate process, a new and effective fuzzy color-based

over-segmentation method is suggested. The second method generates virtual ontology with 3D information using multi-view scenes. The many ambiguities in extracting 3D information are resolved by employing a new fuzzy dynamic programming method (FDP). The hybrid approach of FDP and 3D reconstruction method generates more accurate *virtual ontology* with 3D information.

A virtual model is equipped with virtual ontology whereby contextual knowledge can be mapped into the Metaearth architecture via the proposed isomorphic matching method. The suggested procedure guarantees the automatic and autonomous processing demanded in *virtual interaction analysis* with far less effort and computational time.

To my parents, *Sejong Lee* and *Suna Kim*

ACKNOWLEDGEMENTS

I am heartily thankful to my Ph.D. advisor, Dr. Banerjee. This dissertation and related research studies could not have been completed without the great guidance and continuous encouragement of Dr. Amarnath Banerjee. I will not forget his passion and great personality forever. I would also like to acknowledge my committee members: Dr. Lewis Ntaimo, Dr. James Wall and Dr. Yoonsuck Choe for their encouragement and precious comments.

There are many other people at Texas A&M University who have helped me stay motivated. My lab members, Dr. Bikram Sharda, Dr. Abdullah Cerekci and Hongsuk Park provided insights that guided and challenged my thinking, substantially improving my research product. I am also making it indebted to many of my student colleagues. Additionally, the Department of Industrial and Systems Engineering provides an stimulating environment in which to learn and grow. I am especially grateful to Judy Meeks for always being so kind in assisting me in many different ways.

On a more personal note, I would like to thank my parents, Sejong Lee and Suna Kim, and my brother, Kangsoo Lee and his wife, Dayoung You, for keeping me always positive and supporting my career decisions with their endless sacrifices. Finally, a special thanks to my wife Heejae Choi for her patience and faith in me. Without her support, understanding and love, this dissertation would not have been completed.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

Page

Page

Page

LIST OF TABLES

1. INTRODUCTION

Rapid development of 3D technologies is encouraging wider application of virtual model and many virtual reality (VR) technologies in the design, control, monitoring and simulation of industrial processes. In 3D technologies' early stage, the size of virtual environment-embedded applications in manufacturing was relatively small – from *equipment, workstation*, *cell*, and *shop floor* levels to *facility* levels [1]. Now, the sizes and complexities of virtual systems have grown far larger due to the Internet and ubiquitous technologies.

This trend has combined with large-scale virtual environments (LSVEs), such as *metaverse* [2, 3], *networked virtual environments* (NVEs) [4], *massively multi-user virtual environments* (MMVEs) [5, 6] and *large-scale distributed simulations* (LSDSs) [7]. To date, however, much of the relevant literature has ignored how to effectively generate these types of LSVEs. Especially at issue is the quantity of virtual components that can be quickly and simply constructed using the appropriate data structure. Lee and Banerjee have suggested an architecture and interaction modeling methodology known as the Metaearth architecture [8]. This dissertation extends their work by describing the construction of this new methodology from different input sources.

Since virtual components in LSVE often are derived from components in real applications, key characteristics include an automatic/semi-automatic mechanism and

---

This dissertation follows the style of *Journal of Manufacturing Systems*.

descriptions of the interactions or relationships among the constructed virtual components. To describe any relationship, each virtual component requires a certain type of structures and application-related semantics. The semantics can be represented using several contexts. For example, assume a virtual factory with conveyor belts, robots and industrial machines. The tasks are to visualize the actual factory, as well as measure several production performances against time and space constraints. Each virtual robot needs to be equipped with a structure and contexts in order to produce optimal analyses and simulations. Hence, the structure of each virtual robot will be used for a kinematic structure of the virtual robot, and the mapped contexts of each virtual robot part can be used for the constraints of each component of the virtual robot. Figure 1 shows an example of structure and contexts in a virtual robot.

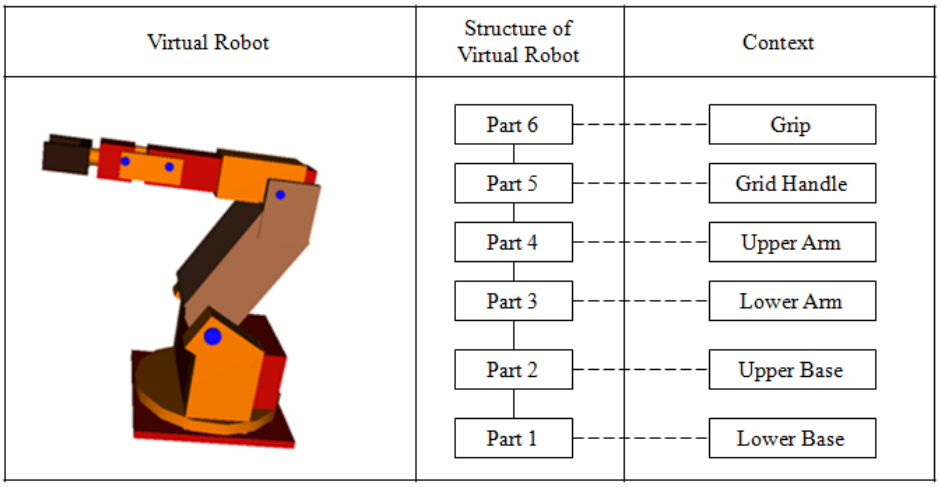


Figure 1. Structure and context in a virtual robot.

In a virtual environment, such as a virtual model of natural scenes, the structure can be represented with a "scene graph". Without this structure and mapped context information, the generated LSVE with many virtual components cannot be used automatically or semi-automatically in the next process: *virtual interaction analysis* – the analysis and simulation activity using virtual models' interactions. Some examples are:

- Traffic simulation such as navigation and collision testing
- Flight simulation
- Driver training
- Electronic simulation
- Video games

As structures and contexts of virtual models have crucial roles in *virtual interaction analysis*, the extraction process of structure and context mapping activities is an important task in the establishment of LSVE.

We caution, however, that knowledge extraction from unknown environments is one of the most challenging issues encountered in science and engineering fields, because knowledge can be a set of meaningful data among overall data, relationships (between the target and source) or inference rules.

Generally, the knowledge extraction procedure consists of a classification process and an intensification process. The classification process identifies the meanings from source data, and the intensification process increases the degree of the meanings. In

the computer science domain, many pattern extraction techniques can be mapped to classification techniques, and learning algorithms for the intensification process. The identified pattern via the classification process can be targeted knowledge itself, or it can be used as the basis for acquiring knowledge. We can combine the patterns with learning algorithms to make them converge with the targeted knowledge. In this fashion, meaningful data can be extracted and converted into knowledge.

Knowledge extraction functions as an important tool for the seamless processing of systems and information. The term, "seamless execution", relates to "autonomous/intelligent execution". Suppose that a process consists of several sub-processes. An initially designed model and input data (e.g., system parameters) may be changed after processing every sub process. Whenever the parameters or data are changed at each sub process, efforts such as a modification of data or parameters may be needed for the next sub process. Such efforts can prove a significant burden by limiting rapid execution and autonomous processing. Yet, if a type of knowledge extraction process between two consecutive sub processes exists, knowledge can then be extracted from a previous sub process and used to reduce the effort required for subsequent sub processes. Furthermore, knowledge extraction is a basic process for constructing intelligence. A system with knowledge extraction functions can greatly increase its degree of knowledge and handle dynamic changes inside and outside its environment.

This dissertation will introduce a more efficient methodology for constructing large-scale virtual environments with *virtual ontology* from 2D images. A hierarchy with contexts is referred to as *virtual ontology* (the detailed definition appears in Section 3).

The overall procedure is termed "*Large Scale Virtual Environment (LSVE) construction through extracting virtual ontology*". An LSVE with *virtual ontology* can be used for *virtual interaction analysis* automatically or semi-automatically. A 3D traffic simulation can be an example of a *virtual interaction analysis*, since 2D photos gathered from multiple cameras or active vision systems along a road cannot describe a car accident in full detail, due to hidden or untaken features.

The dissertation is organized as follows: Section 1.1 provides an overview of the methodology and introduces the problem statement. Section 2 describes the background and related research studies. Detailed definitions of virtual model, context, ontology and their relationships with the effective data structure for LSVE are given in Section 3. Section 4 describes the construction of LSVE using one 2D image and Section 5 describes the construction process using multi-view scenes. Sections 6 and 7 describe a context mapping process for LSVE and related issues.

1.1. OVERVIEW OF PROPOSED METHODOLOGY

This dissertation focuses on the generation of a virtual environment with *virtual ontology,* using the Metaearth architecture [8] as a data structure for describing relationships among virtual components and capturing common patterns in virtual components. Figure 2 illustrates the methodology.

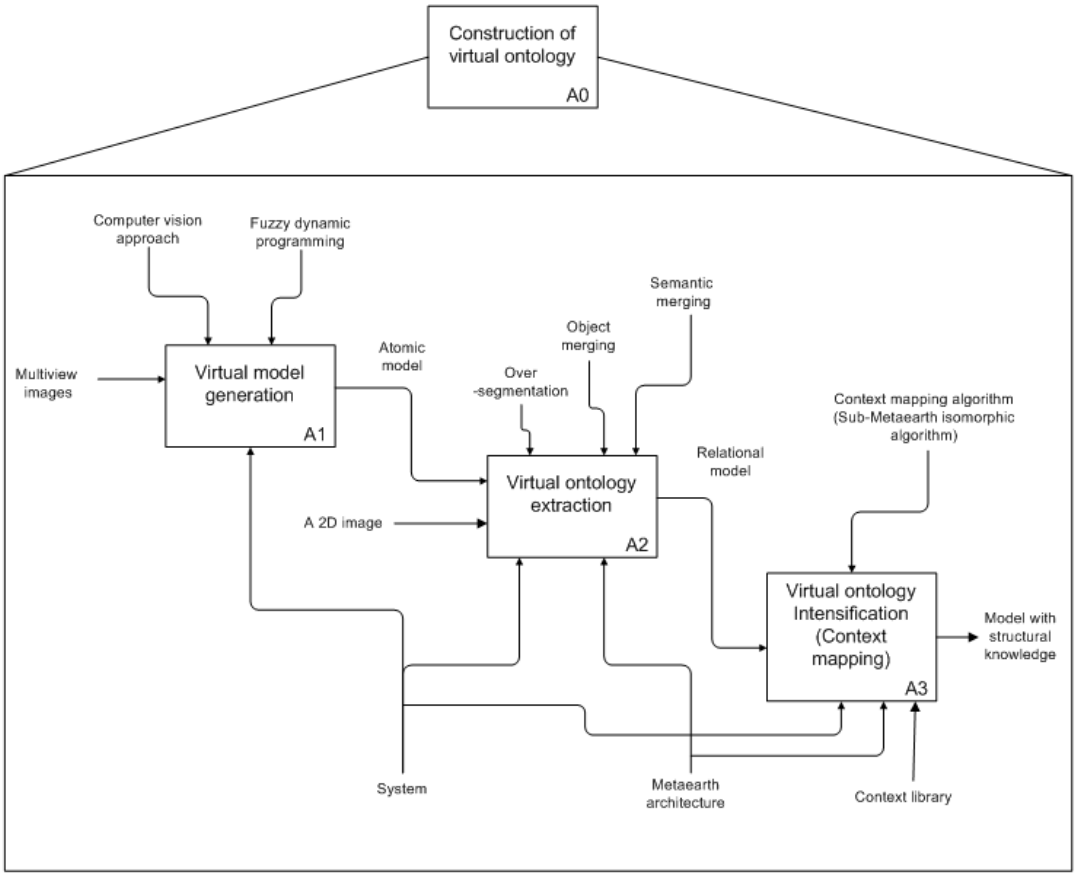


Figure 2. IDEF-0 diagram of the suggested approach.

(*This IDEF-0 diagram is made by the A10 WIN, KBSi*)

The Metaearth architecture can be generated in two ways. The first method uses one 2D image while the second uses multi-view 2D images. The first method is related to scene decomposition or scene understanding techniques in the existing literature. Scene understanding can be summarized as "a 2D scene without any structure and context is analyzed and merged into some related groups". Figure 2 shows an IDEF-0 diagram (level-1) of the first method.

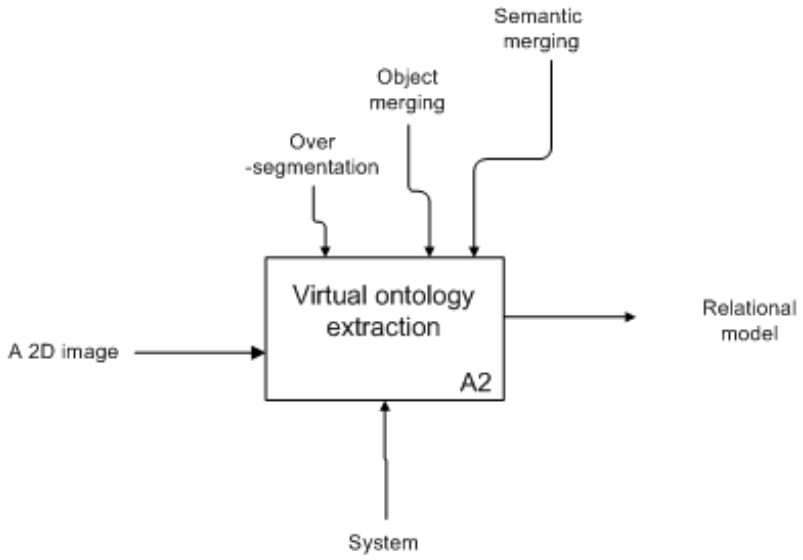


Figure 3. IDEF-0 diagram of "*virtual ontology* extraction from a scene".

This process is called "*virtual ontology* extraction from a scene" in this dissertation. Figure 3 shows IDEF-0 diagram for "virtual ontology extraction from a scene". As an input, a 2D scene such as a photo or a sketch is considered and it is converted into *relational model*. In this dissertation, a virtual model or generated architecture is defined as an *atomic (virtual) model*, *relational (virtual) model* and *model with structural knowledge*, as follows:

Definition 1. *Atomic (virtual) model*

A (reconstructed virtual) model which does not have any relationship among model components. For example, a reconstructed 3D model using general stereo vision techniques is an atomic model.

***Definition 2**. Relational (virtual) model*

A (virtual) model with relationships among model components which can be represented as a graph, structure, hierarchy or architecture. For example, a 3D model with its scene graph is a relational model.

***Definition 3**. (virtual) model with structural knowledge*

A model with a hierarchy among model components and contextual knowledge in the hierarchy which can be used directly in virtual interaction analysis and can also interact with other models with structural knowledge.

Figure 3 (and Section 4) illustrates the generation of a *relational model.* Since its only input is a 2D image, Z depth cannot be extracted in general. Thus this model is a "*relational model*" with a hierarchy. Like other scene understanding techniques, the suggested technique uses "over-segmentation", yet the technique differs, because it converts a 2D image into a type of graph structure and modifies it. The detailed procedures and explanations are discussed in Section 4.

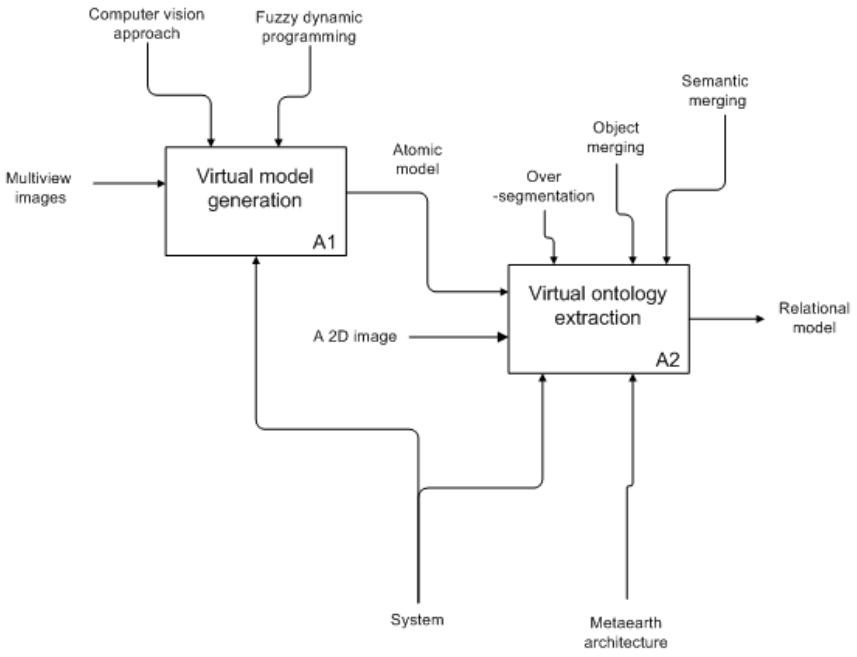


Figure 4. IDEF-0 diagram for generating a virtual model with *virtual ontology*.

The second method generates Z-depth information using stereo vision or multi-view vision approaches, producing a hierarchy with X, Y, Z coordinates. Figure 4 illustrates an input-output-control-mechanism (ICOM) model of this approach. The detailed processes and implementation are discussed in Section 5.

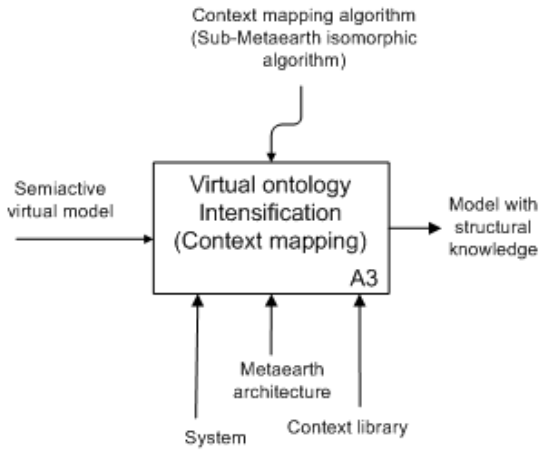


Figure 5. IDEF-0 diagram for context mapping procedure.

Using both methods, we can now generate a *relational model*. The remaining procedure is to map contexts into the obtained hierarchy. This objective is achieved using comparisons between the generated Metaearth architecture and context library. Since both the generated Metaearth architecture and context library are represented as types of graphs, we classify the comparison as a graph-subgraph isomorphic matching problem [9]. Figure 5 shows the ICOM model of the context mapping procedure. The difference between a general subgraph isomorphic matching and this problem and an algorithm for mapping context using the Metaearth architecture is explained in Section 6.

2. BACKGROUND AND LITERATURE REVIEW

This dissertation incorporates knowledge and issues in areas such as computer vision, pattern classification, optimization, data modeling and semantics/ontology. As a result, many theories and implementations are required. In general, state-of-the art research studies term *virtual ontology* extraction (A2 shown in Figure 2), a scene understanding, image parsing and decomposition.

Table 1. Related research fields and usage.

Related field and algorithm	Usage
Computer vision algorithm	- 3D reconstruction
Pattern classification algorithm	- 3D reconstruction - Construction of virtual model - Reorganization of *virtual ontology*
3D modeling and geometric modeling	- Representation of virtual model - Extracting corresponding / related data
Optimization methods	- Fast reconstruction - Occlusion handling - Construction of virtual ontology - Reorganization algorithm
Fuzzy logic and control	- Ambiguity handling in 3D reconstruction - Ambiguity handling in pattern classification
Graph theory	- Graph isomorphic matching - Graph-subgraph isomorphic matching
Data / information / knowledge modeling	- Modeling hierarchy for virtual model - Representation of semantics - Modeling *virtual ontology*
Virtual analysis and simulation	- Verification of effectiveness of generated virtual model and *virtual ontology*

The virtual model generation procedure (A1 in Figure 2) is related to many 3D reconstruction techniques, and virtual ontology intensification (A3) is related to ontology/semantics related studies. Table 1 shows several related research fields and stages in which the various algorithms are used. The following sections classify the related research studies by 3D reconstruction (A1), scene understanding (A2) and context mapping in a graph structure (A3).

2.1 RESEARCH STUDIES FOR VIRTUAL MODEL GENERATION

Existing 3D reconstruction algorithms have been classified according to these criteria:

- System input and source type
- Preprocessing criteria
- Modeling methodologies
- Post-processing task
- Types of virtual model
- Format of virtual model
- Usage/application of virtual model

- Format of ontology/knowledge

Figure 6 shows a classification for generating a virtual model using existing algorithms which can be used for both identifying the characteristics of each algorithm and comparing the algorithms.

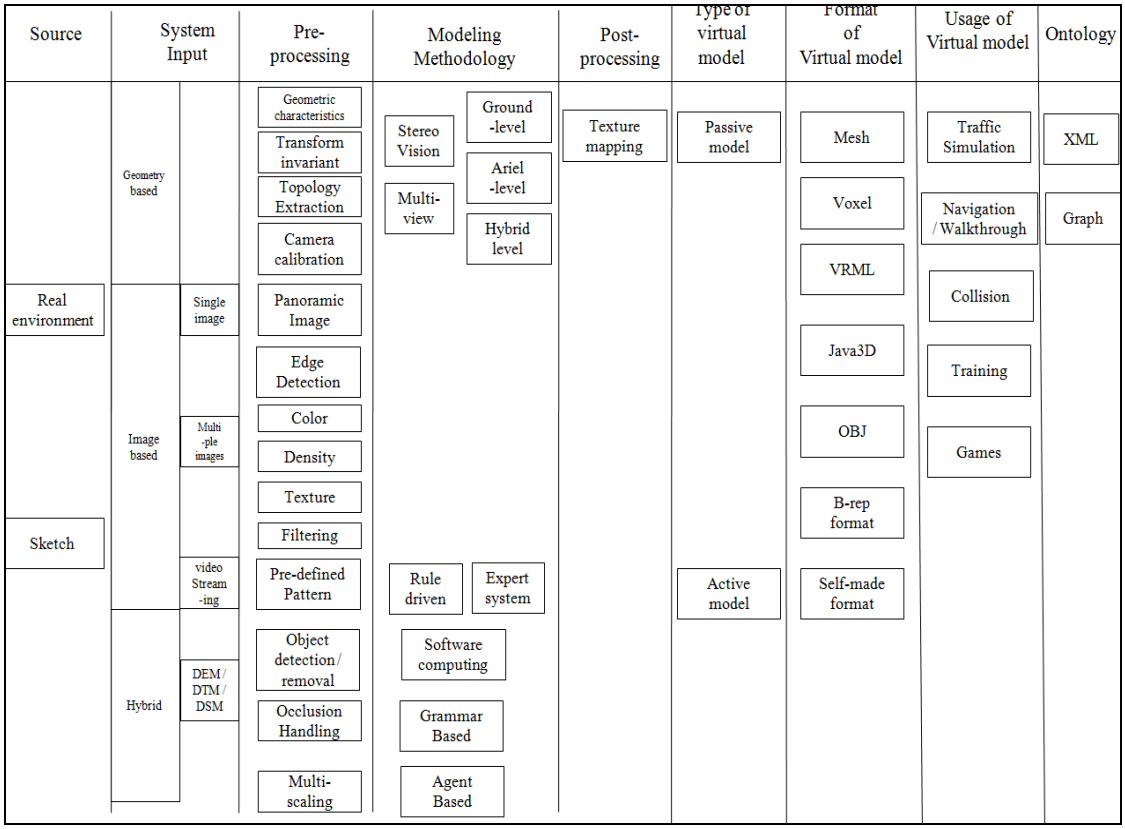


Figure 6. Classification of existing methodologies.

For example, Cornelis *et al*. reconstruct a 3D urban scene using stereovision matching [10] with images from a streaming video. The model has been used for identifying vehicles on the road. Their algorithm can be classified based on the blue procedures in Figure 7; many existing research studies using stereovision have similar frameworks.

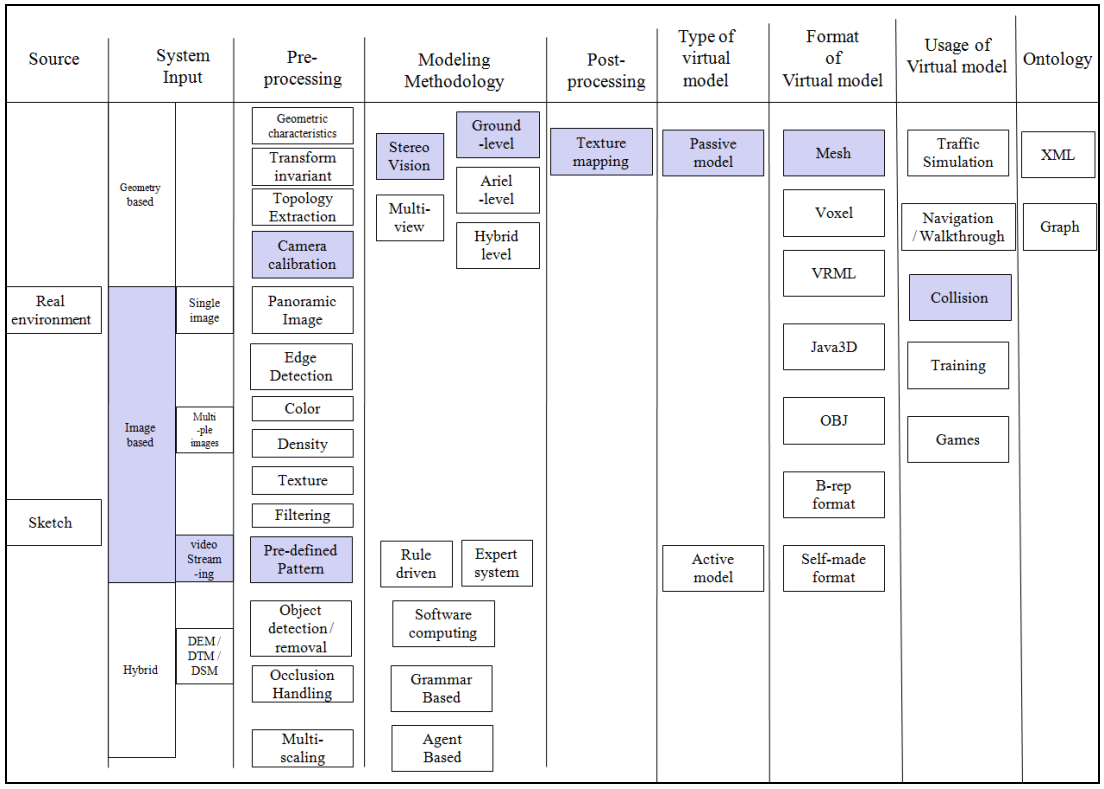


Figure 7. Cornelis *et al*.'s method [10].

The main characteristic of stereo vision-based 3D reconstruction is that the generated model is usually an *atomic model* (see Section 1.1), meaning that it has no

hierarchy. This limitation causes difficulties for the generated virtual environment's uses in a *virtual interaction analysis*.

Other algorithms can construct a *relational model* or *model with structural knowledge* having *virtual ontology*. In general, an *active virtual model* can be generated using expert systems or pre-defined rules. Grimsdale and Lambourn employ an expert system to identify types of roads [11] using parameters such as average length, curvature, width, junction type and so on. Even though the generated virtual road model is an *model with structural knowledge*, the usage of parameter-based identification methods causes low effectiveness and fails to identify more complex shape and models. This approach is in blue in Figure 8.

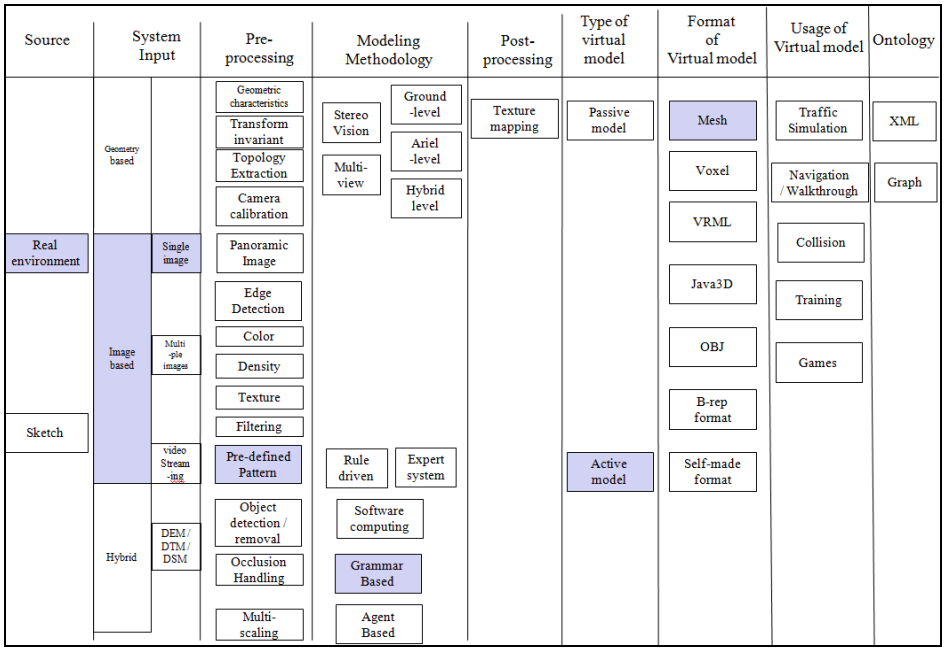


Figure 8. Grimsdale and Lambourn's reconstruction method using an expert system [11, 12].

Lechner *et al*. use an agent-based system to describe a city model [13]. However, these algorithms are unsuitable for generating a LSVE. To overcome this limitation, a vision based virtual model generation approach is described in section 5.

Input data and recognized data can be considered as criteria, such as:

- Geometry-based construction
- Image–based construction
- Hybrid construction

Geometry-based algorithms construct a 3D model from point clouds. The points are measured and usually generated using 3D scanners. These methods are reviewed in Biggers *et al*. [14] in detail. The main characteristic is that these points already contain depth information (e.g. Z values). Using this depth information, a mesh model can be generated and converted to a 3D model format such as B-rep format. However, these methods are also not suitable for reconstructing LSVEs such as buildings or cities.

Another approach uses 2D images as an input. The images can be stereovision/multi-view based images, sketches or other general images. At source level, these images can be classified into ground-based imagery, airborne imagery or hybrid imagery. The level information of input images strongly influences the shape and characteristics of reconstructed 3D volume. For example, ground-based imagery methods create a 3D virtual model without roof information. Some research studies have used hybrid approaches considering both geometry information and image information, i.e. *Light Detection* and *Ranging* (LiDAR) data [15-17]. With similar data,

digital elevation model (DEM), *digital terrain model* (DTM) and *digital surface model* (DSM) have been used [18, 19]. These data have Z-depth information, making them, useful for acquiring more accurate virtual models and generating large scale virtual models. However, these methods are dependent on the measuring devices.

Regarding the issue of data format, Prusinkiewicz *et al*. have introduced L-system and the Chomsky grammar to describe plants' shape [20]. While their data format can track changes in design, it is only useful for 2D models and cannot describe complex 3D volumes.

The suggested approach in Section 5 generates a virtual model from several 2D images overcoming these limitations. The algorithm uses computer vision techniques and fuzzy logic for handling ambiguities in 3D reconstruction. The details are provided in the following sections.

2.2 RESEARCH STUDIES FOR SCENE UNDERSTANDING

One research trend is a new and important topic in image processing. It is known by similar names, such as: scene understanding [21, 22], generation of scene structure [23], scene decomposition [24], image parsing [25], image labeling [26, 27], and multi-class image segmentation [28-30]. The main objective is to obtain a scene structure from an input image, such as a taken photo or multiple images such as image frames or stereo

images. The acquired structure is related to the knowledge acquisition process, as discussed in Section 1. The research is relatively new and there are fewer examples comparatively than other image processing techniques.

An early stage, however, has been invoked by studies in image segmentation [31-39]. Using visual cues, extracted edge information and shape information, an image is first over-segmented and the over-segmented regions are then merged using pre-defined similarity measures. Since these image segmentation techniques fail to describe the meaning of each reason and relationship between two regions, additional research examines efforts for representing relationship among segmented regions and generating a structure. Various pattern classification techniques are applied and merged with image segmentation techniques. For example, some predefined patterns are detected using detection algorithms and the detected image regions are converted to a structure. Tu *et al*. [25] detects faces and texts from a image using a Bayesian approach and *AdaBoost* method. However, their approach can parse only the pre-defined patterns.

Another avenue of research uses color, shape [40] or other information [41, 42], to first over-segment an image and merge it using supervised learning methods. Gould *et al*. [23] use a specific energy function for merging over-segmented regions and train it using predefined patterns such as sky, tree, road, grass, etc. In general, this approach depends heavily on the quality of the trained model and input data. This characteristic has a significant influence on the performance of each algorithm. Another characteristic of these research trends is that the output of these algorithms is only segmentation. Even though grouped segmentations may have several meanings, they are not useful for

capturing and reorganizing higher semantics. The limitations and disadvantages of current scene understanding approaches are summarized as:

- Heavily dependent on predefined patterns and training algorithms: most approaches use supervised learning techniques
- Most over-segmentation techniques use a visual cue such as color and cannot generate more accurate structures
- 3D information is absent
- Most data formats of the generated structures are only segmented regions and insufficient for representing a scene structure

Recently, Saxena *et al*. [21] obtained a 3D scene structure from a single still 2D image using *Markov Random Field* (MRF) which is trained by supervised learning technique. Even though their method generates 3D models for 64.9% of 588 images, its performance depends on the trained MRF model. Z depth information is also driven from the trained MRF model. Following Saxena *et al*., Delarge *et al.* [43, 44] and Hoiem *et al*. [24, 45] assume that the environment is made of a flat ground with vertical walls, but this assumption generates an inaccurate structure.

Table 2. Characteristics of existing research studies in scene understanding.

Research studies		Learning method	Format of scene structure	Generation of 3D information
Scene Under-standing	Saxena *et al*. [21]	Supervised Learning (Bayesian method)	MRF	O
	Delage *et al*. [43, 44]	Supervised learning	segmentation	O
	Hoiem *et al*. [24, 45]	Supervised learning	segmentation	O
	Tu et al. [25]	Bayesian methods	segmentation	X
	Gould et al. [23]	Supervised learning	segmentation	X

The suggested algorithms in Sections 4 and 5 overcome the limitations listed in Table 2. The scene structure can be represented as a special graph structure (the Metaearth architecture in Section 3) and it can be generated from one scene (the methods in Section 4) and multiview images (the methods in Section 5). The suggested algorithms, presented in Section 4 and 5 do not depend on any supervised learning method, making them useful in more general application areas. Detailed theories and discussion are described in the following sections.

2.3 RESEARCH STUDIES FOR ONTOLOGY AND SEMANTICS

A *model with structural knowledge* has a hierarchy and semantics (Definition 3). In this dissertation, the semantics-mapped hierarchy is called a *virtual ontology* (the detailed definition is described in Section 3.1). In the *model with structural knowledge*, semantics are represented as contextual knowledge which is created by combining virtual components in the generated structure and context.

Montague and Dowty [46] first introduced and developed semantic theory in research fields handling natural language. According to their definition, the elements of symbolic objects can be called semantic features and semantics can be a meaning. However, many researchers define ontology according to their own concepts. For example, Anderson and Vasilakis [47] define it in terms of geometric modeling as a rigorous conceptualization of some knowledge domain. By structuring knowledge from a defined ontology, systems can semantically inter-operate when processing, exchanging or presenting information. In this concept, ontology can vary by domains and application areas.

Currently, efforts to extract ontology and semantics are a major topic of research in artificial intelligence, data mining and communication theory [48-51]. In particular, semantics and ontology play a crucial role in the development of semantic Web and Web 2.0 especially the Web Ontology Languages OWL and Resource Description Framework (RDF) [52-54].

In general, the research consists of three streams: how to generate ontology; effective data structure; and how to intensify ontology and semantics. The first stream

can be handled differently in each application. As stated above, this dissertation examines its generation from a 2D image or multi-view images.

The second stream, data format of ontology, is related to the representation of ontology. The most popular format uses a graph-based format or its own language. Goh *et al*. [55] use their own language called COINL via the *Frame Logic* method. While this kind of format is useful for a specific application, it has a disadvantage in general usage and representation simultaneously. Wu *et al*. [56] use a directed labeled graph to represent ontology. The graph is defined using a 4-tuple: concept node, edge for concept nodes, labels for concept node and labels for edge. The main limitation of their ontology graph is that it fails to define relationships among labels. The relationships among labels as well as among conceptual nodes are important in establishing detailed ontology. For this reason, we use Metaearth architecture as described in Section 3.

3. VIRTUAL ONTOLOGY AND METAEARTH ARCHITECTURE

This section presents the exact definitions of *virtual ontology,* Metaearth architecture [8] and related terms used in this dissertation. Section 3.1 explains the concept of *virtual ontology* and its definition in comparison with other formats. Section 3.2 describes the features of Metaearth architecture and gives a summary of Lee and Banerjee [8].

3.1 VIRTUAL ONTOLOGY

As discussed in section 2.3, the definition and concept of ontology varies by application and domains.

In general, ontology is defined by 5-tuple: $O := \{C, R, H^c, rel, A^0\}$. O indicates "ontology", C means "concept", R represents "relation", H^c indicates "concept hierarchy", rel means "function relation", such as relationship between concepts ($R : C \times C \rightarrow R$) and A^0 represents "axiom", such as the relationship between C and R. In this definition, the set $\{C, H^C\}$ is called "*Taxonomy*" and the set of *Taxonomy* , R , rel and A^0 is called "Ontology". However, the general representation is not suitable for *virtual ontology*. We apply ontology to a virtual model or a generated model after

scene understanding procedures. In our concept, ontology is termed *virtual ontology* and defined as:

Definition 4. *Virtual ontology*

A virtual ontology is a structure of virtual environment with knowledge. The structure can be a hierarchy or relationships among virtual components. The knowledge of overall environment or a knowledge belongs to one or some virtual components. The knowledge is embedded in the structure with various formats.

Knowledge represented with contexts is called a contextual knowledge. In this dissertation, knowledge implies a contextual knowledge. A context is a label with one or more meanings. As shown in Figure 1, each part of a virtual robot has a label, such as basis, arm or grip, which indicates that part 1 is being used as a "Lower Basis" of the robot and part 4 is being mapped as a grip. The label defines the usage of a virtual component and guides its action. We call the label a context, and a type of knowledge in a virtual environment is represented using contexts.

Since our objective is to generate a virtual environment's hierarchy and its contextual knowledge, the identification of all virtual components is the prior condition for establishing *virtual ontology*. Once the virtual components are detected, we can generate the structure, which can be a kinematic structure, a structure describing interaction, a hierarchy indicating distance, or a type of scene graph.

Figure 1 shows that the virtual robot's structure is represented by its kinematic structure. When a mechanical part or assembled products are represented in a virtual model, it is reasonable to consider a kinematics as a structure. When using a virtual environment to describe interactions between virtual objects, the structure must indicate the degree of interaction, i.e. high and low. In most games or simulations using virtual components, the overall structure represents interactions among inner components. In this manner, the structure can be altered by the characteristics and objective of the virtual environment.

However, in a generic image such as a natural scene or photo, it is quite difficult to capture the characteristics or objective of the reconstructed virtual environment. In this case, we can use a type of scene graph as a structure. Sowizal *et al*. [57] define scene graph as a description of a 3D world rendered by a graphic. The format most often used is the directed acyclic graph (DAG) with nodes and arcs of various types. In general, commercial graphics software systems like *VRML*, *OpenInventor*, *Performer*, *Java3D* and *X3D* use this scene graph concept as a structure of overall virtual environments. In the scene graph, arcs represent relationships among nodes using parent-child relationship and reference relationship. With two types of nodes (group node and leaf node), a volume, geometry or texture can be represented. Figure 9 illustrates a scene graph in a *Java3D* model.

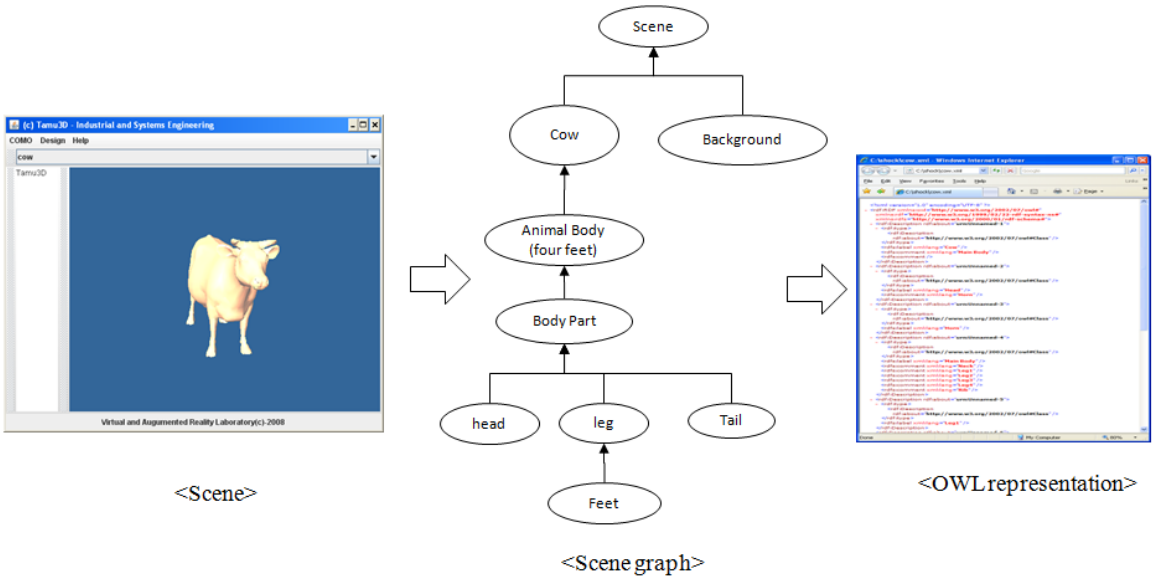


Figure 9. A scene graph and the OWL representation in a *Java 3D* model.
(This Java 3D model is constructed using TAMU3D software which is developed using JAVA3D and VRML)

While a scene graph is a useful description of a virtual environment for a general architecture, its limitations are:

- When a group or patterns are repeated, the scene graph may be complex.
- The context of a virtual component can be represented as only annotation or notes.
- It is not suitable for describing interactions between virtual components.

A more effective structure for describing *virtual ontology* is Metaearth architecture [8]. We propose this architecture to describe any LSVE with many

interactions among virtual components. It allows us to construct an LSVE with less redundant designs. The next section summarizes the characteristics and advantages.

3.2 METAEARTH ARCHITECTURE

"Metaearth" is a synthetic word combining *meta* and *earth* [8]. The architecture itself derives from the earth's structure which consists of an outer core and an inner core, a mantle layer and a crust layer. The crust layer corresponds to the virtual space layer where the virtual components exist. The core layers correspond to the library layer and ontology layer. These layers are "in charge" of the assets of virtual components in the virtual space layer. The mantle layer corresponds to a mapping layer which describes the mapping from shared components in the library layer to virtual manufacturing components in the virtual space layer. Table 3 and Figure 10 describe the roles and components of the layers.

Table 3. Each layer's role and components in Metaearth architecture [8].

Layer		Components	Role
Virtual space layer (Crust layer)		Each virtual object elements	Describe virtual environment to communicate with users
Mapping layer (Mantle layer)		Mapping (replication) from virtual object in core layers to virtual object in crust layer	Each mapping describes rotation angles, (x,y,z) position coordinates and scale vectors.
Core layer	Library layer (Outer core layer)	Outer core layer	Layer for reusability
	Ontology layer (Inner core layer)	Inner core layer	Describes configuration and characteristics of LSVE

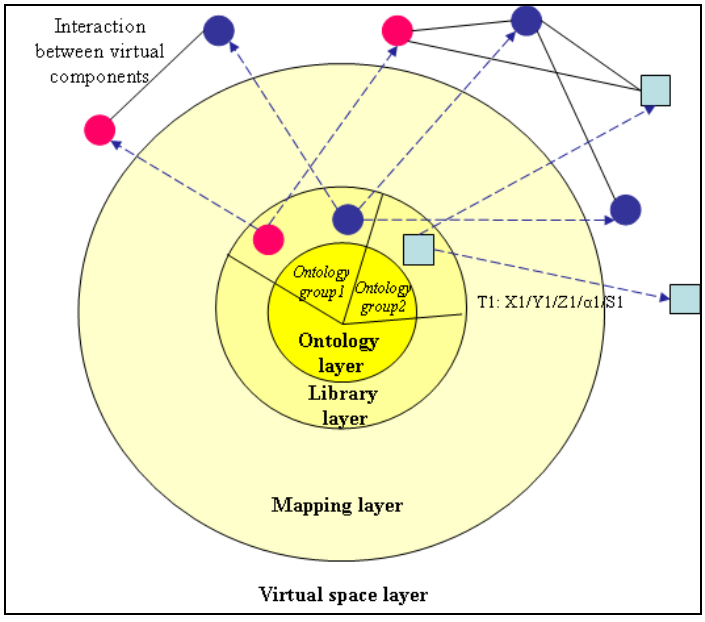
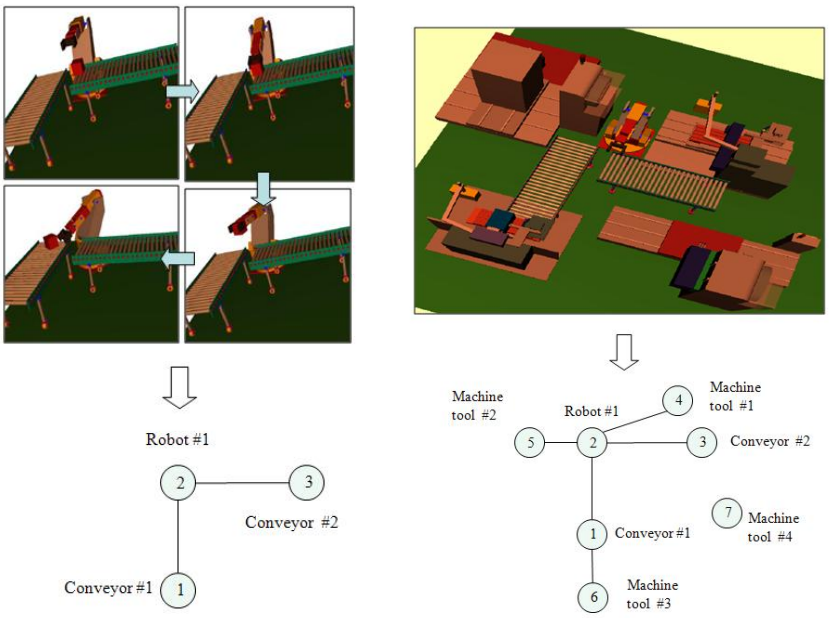


Figure 10. Metaearth architecture [8].

The virtual space layer consists of multi-interactions. This graph's vertices represent virtual components in the virtual environment. These components are replicated from the library (outer core) layer's components. The library layer's role is similar to the library/asset of virtual components used in the virtual space layer. Each virtual component in the virtual space layer may have interactive relationship(s) with other virtual components. This characteristic is useful for describing a LSVE. In a natural scene, this interaction relationship can be replaced by adjacent information for a scene graph. Figure 11 illustrates the interaction among some virtual components. The constructed virtual space layer is a workstation in a manufacturing facility.



(a) Interaction representation at machine level

(b) Interaction representation in manufacturing cell

Figure 11. Virtual space layer's interaction representation [8].

(This simulation is performed using *virtualFactory* software which is developed using *JAVA3D* and *VRML*)

In Figure 11 (a), the industrial robot loads products from one conveyor belt to the other conveyor belt. It can be interpreted as the industrial robot communicating/interacting with the two conveyor belts. The relationship is represented using the interaction graph shown below the snapshot in Figure 11 (a). Next, Figure 11 (b) provides the interaction graph for a small manufacturing cell, where conveyor belts #1 and #2 are modeled using one conveyor belt model. The original model of the conveyor belt is defined in the library layer. It is replicated in two conveyor belts in the virtual space layer. This approach increases the reusability of virtual models. Figure 12 shows this mapping; the information in the mapping layer contains the rotation axes and angles, center positions' coordinates and scale vector. This description is represented using XML format for distributed configuration between multi-virtual space layer and library layer.

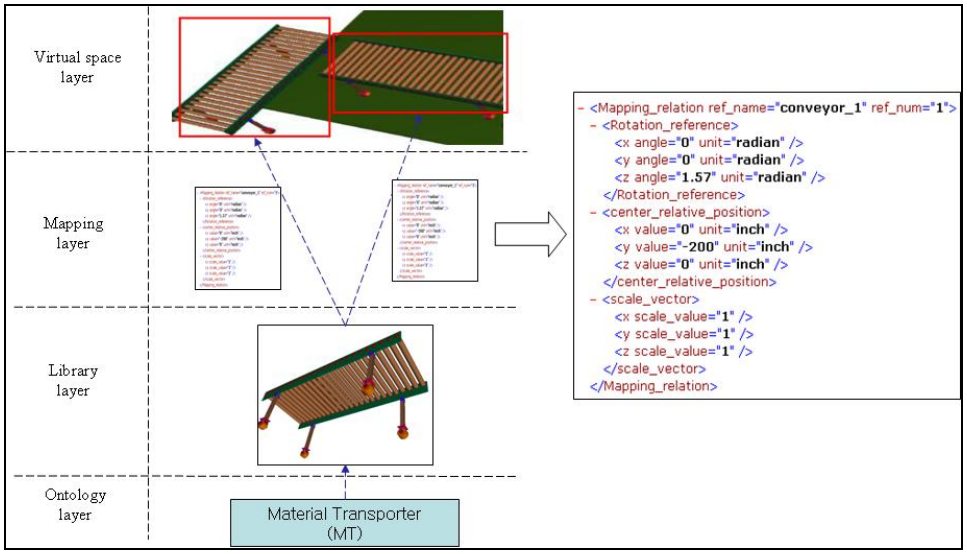


Figure 12. Metaearth architecture for conveyor belts [8].

Metaearth architecture is useful from the perspective of high reusability of virtual components and allows us to check the structure of the virtual environment. The architecture has interaction relations or hierarchical relationships in the virtual space layer and ontology information in the ontology layer. Mapping between the two layers occurs in the mapping (mantle) layer using XML format.

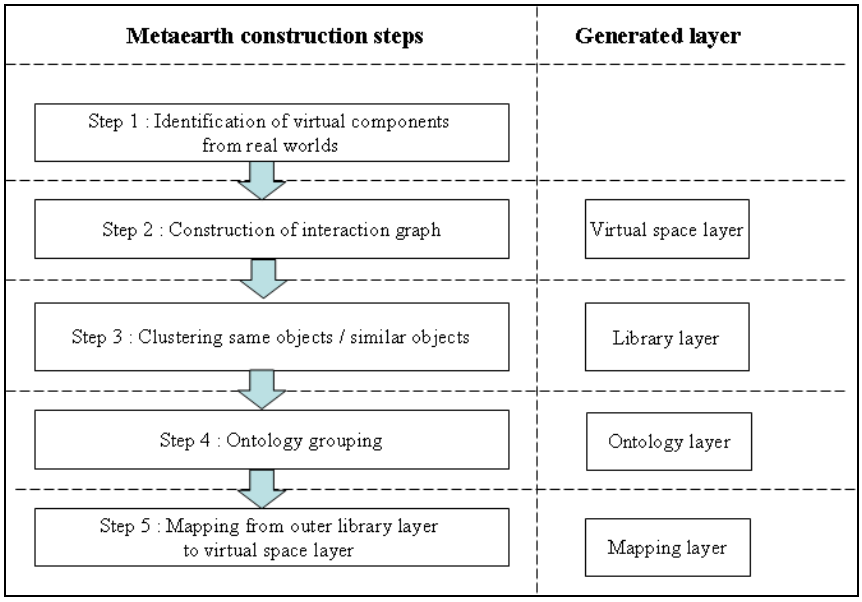
Note that we can expand the architecture to many large-scale environments. In this case, the virtual space layer can be divided into multi-interaction graphs. The library layers as a virtual models' library can be divided into more detail levels and sub-layers. Each mapping is accomplished by checking the ontology.

Metaearth architecture has eight characteristics:

- Reusability
- Support for large-scale virtual environments
- Scalability
- Collaboration
- Distributed control
- Dynamic reconfiguration
- Interaction management
- Support for a variety of industrial areas and applications

Table 4 shows the manual Metaearth construction procedures. One objective of this approach is to generate the Metaearth architecture automatically (achieved by the suggested approaches in Section 4 and 5).

Table 4. Metaearth construction procedures and generated layer in each step [8].



In the virtual space layers, the same component and similar components are classified. Through this process, we extract the common components and use them for virtual model library/assets. Even though one virtual component is not shared with any other virtual component, we can use it as a reference model in the next LSVE. We can also expand Metaearth architecture to a significantly larger LSVE. Figure 13 illustrates the distributed virtual plants concept [8] where the distributed facilities are established effectively with the shared components via the use of Metaearth architecture.

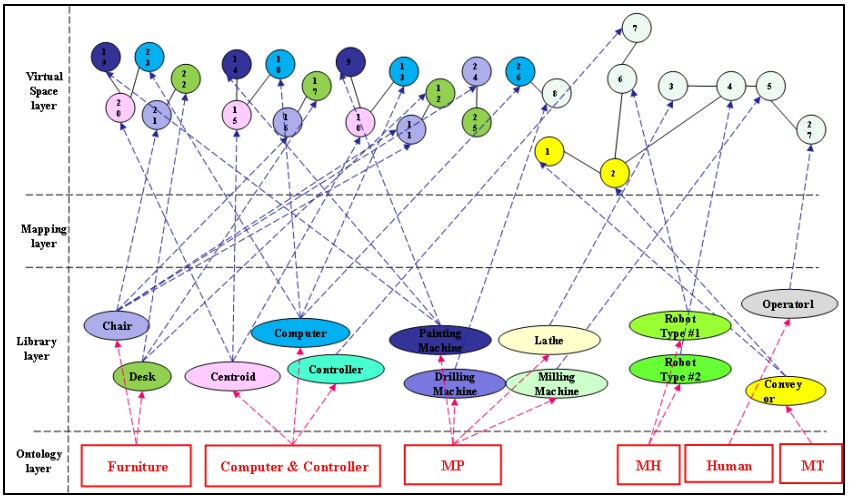
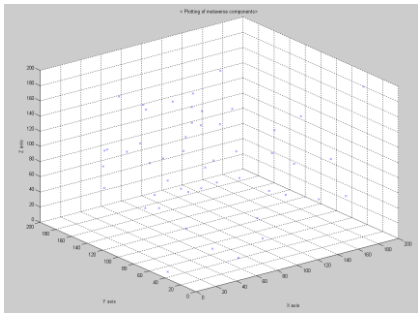
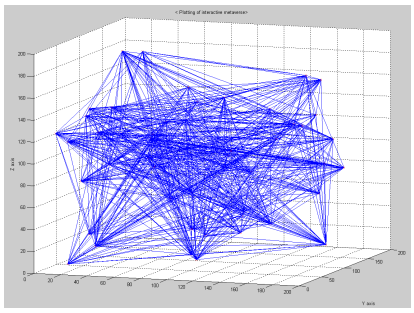


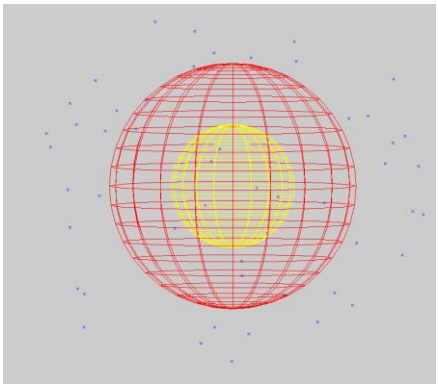
Figure 13. Metaearth architecture for a virtual factory [8].



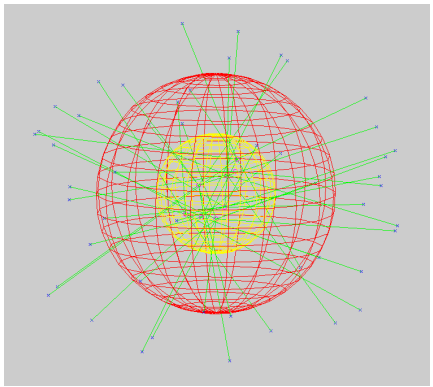
(a) 50 plants in virtual space



(b) Interaction graph among 50 plants



(c) Generation of library layer



(d) Generation of mapping layer

Figure 14. Generation of each layer in Metaearth architecture [8].

50 arbitrary 3D points are randomly generated as the locations of 50 plants (Figure 14 (a)). The interaction or adjacent relationship is generated as shown in Figure 14 (b). These interactions among 50 virtual components are randomly generated. We consider 8 components as shared virtual objects in the library layer. Based on this situation, we create the library layer and mapping layer as shown in Figure 14 (c) and (d). Finally, Figure 15 shows the multi-distributed virtual plant model using the Metaearth architecture (note that we have removed some of the interactions for a clearer understanding).

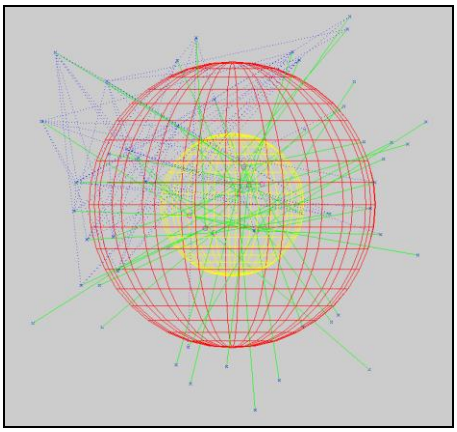


Figure 15. Metaearth architecture for distributed virtual plant model.

This architecture is considered as a good data structure keeping *virtual ontology* defined in section 3.1. As it has virtual components, their relationship, ontology / library and their mapping, various relationship and semantics can be represented in the Metaearth architecture.

Definition 5 defines *virtual ontology* using the Metaearth concept:

Definition 5. *Mathematical definition of virtual ontology*

Virtual ontology is defined as a graph:

$$VO := \{VC, I, C^1, MR, C^2, CR\}$$

Where, VO is "virtual ontology",

VC is the set of virtual component ,

I is the interactive relationship such as $I : VC \to VC$ *,*

C^1 *is the set of shared components,*

MR is the mapping relationship between C^1 *and VC such that* $MR : C^1 \to VC$ *,*

C^2 *is the set of context,*

CR is the relationship between C^2 *and* C^1 *such that* $CR : C^2 \to C^1$

In definition 5, VC is a set consisting of a virtual space layer in the Metaearth architecture. C^1 and C^2 are the components consisting of the library layer and ontology layer. Finally, the context C^2 is mapped to C^1 using CR and C^1 is linked to each VC through MR. From this viewpoint, MR can be considered as a component of the mapping layer in Metaearth architecture. Virtual ontology can be represented using this Metaearth architecture. The next sections explain how we can automatically or semi-automatically generate Metaearth architecture from scenes.

4. CONSTRUCTION OF VIRTUAL ONTOLOGY USING ONE SCENE

4.1 OVERALL PROCEDURE FOR VIRTUAL ONTOLOGY GENERATION FROM A SCENE

This section describes how to generate Metaearth architecture with *virtual ontology* from a 2D image. The process starts by over-segmenting a scene (the 2D image) using the proposed algorithm. Recall that the suggested approach generates relationships between components which are not adjacent as an additional advantage.

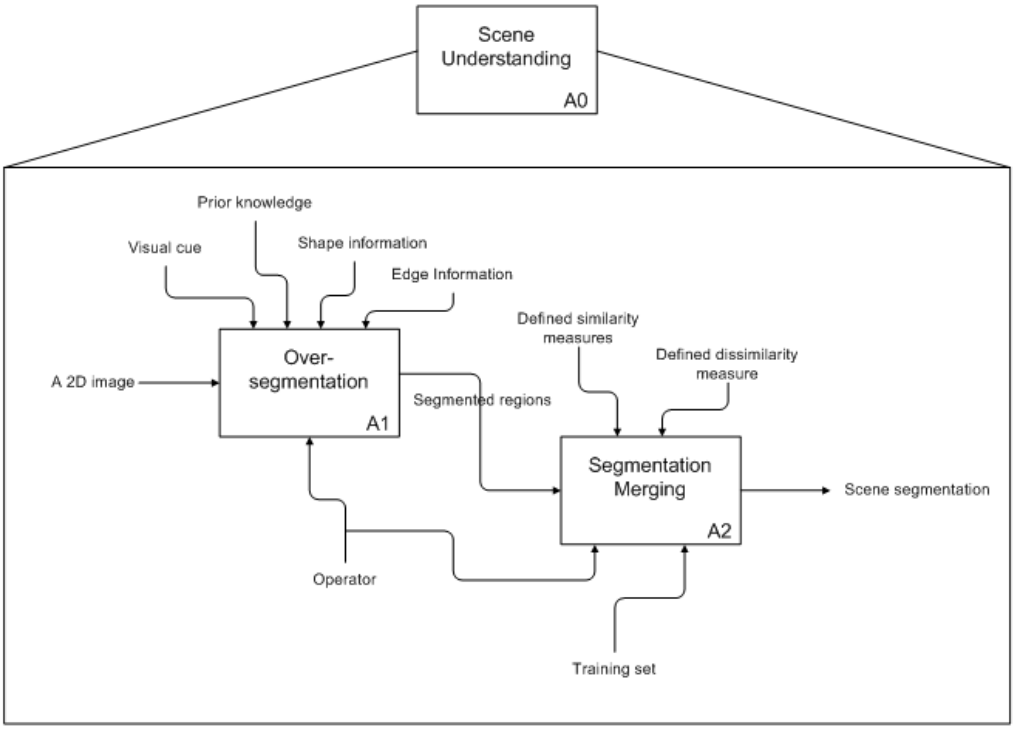


Figure 16. A general procedure of existing scene understanding algorithm.

Next, we group the segmented regions via the object merging process and the semantic merging process. Finally, the 2D scene is converted into Metaearth architecture (described in Section 3.2). Figure 16 shows the general procedures of existing scene understanding techniques using IDEF 0 diagram and Figure 17 illustrates the procedure broken down by steps and sub-steps.

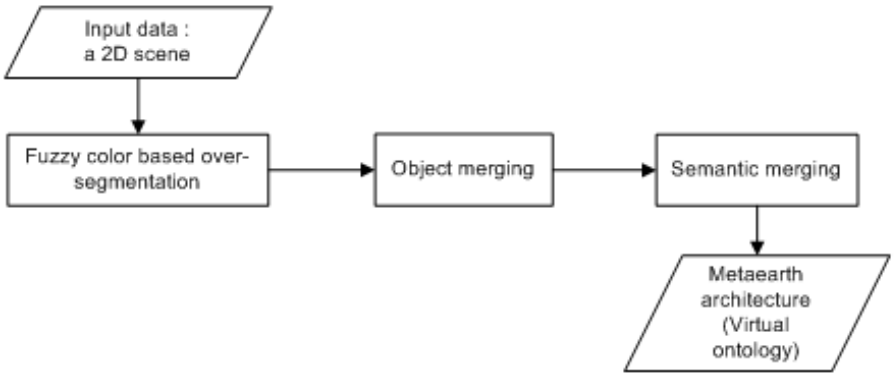


Figure 17. The general procedures for generating virtual ontology from a scene.

The three sub-steps depicted in Figure 17 and detailed in Figure 18 are: 1) fuzzy color-based over-segmentation; 2) object merging; and 3) semantic merging.

1) Fuzzy color-based over-segmentation

The approach considers over-segmentation as a pre-processor which is dependent on prior knowledge or training mechanisms. Most methods such as Chung and Fung [58] use a pre-defined fuzzy color for segmentation, but the result and quality are not stable,

because uniform color quantization or trained fuzzy color are not extracted from an input.

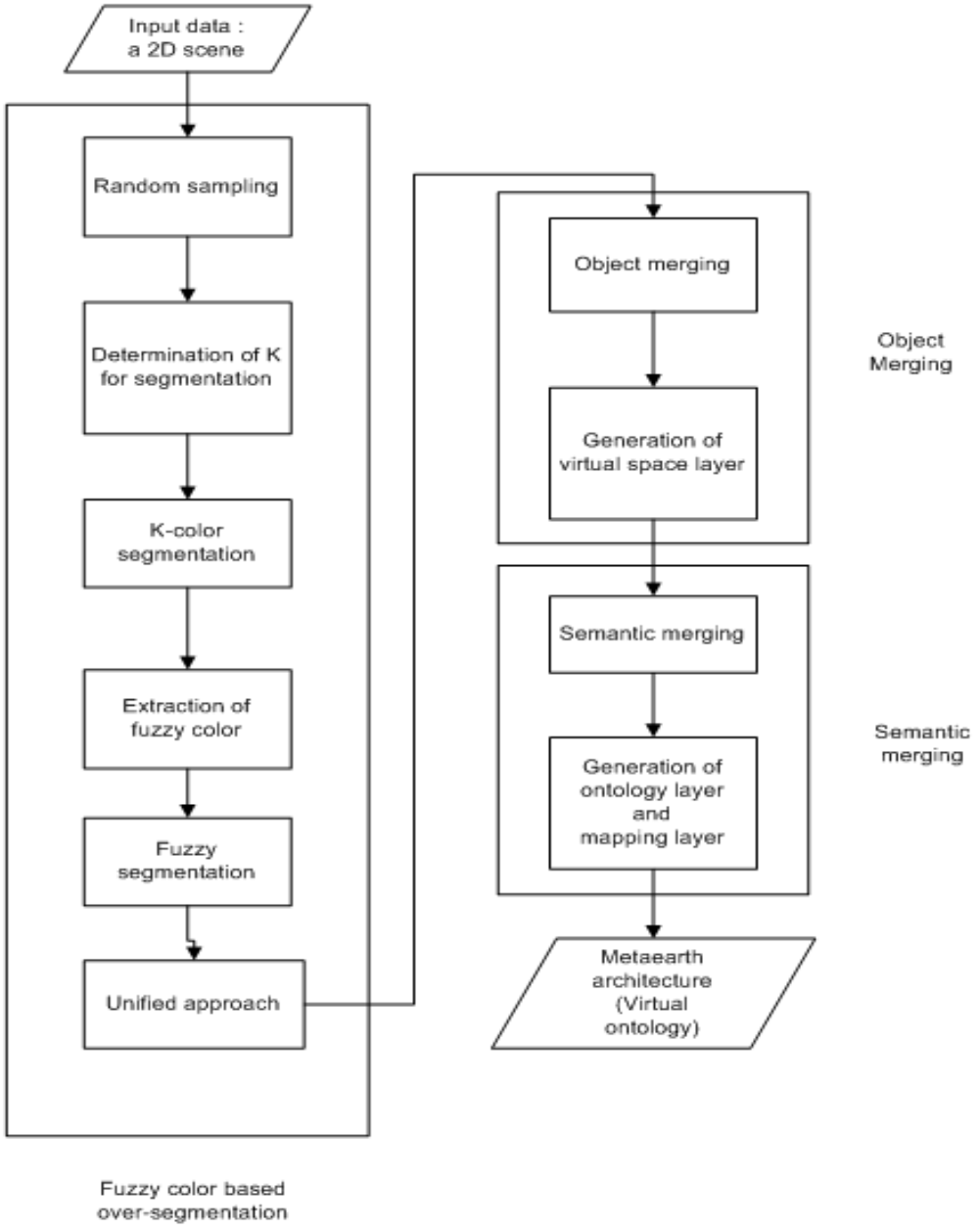


Figure 18. Detailed sub-processes of virtual ontology generation from a scene.

2) Object merging

The segmentation process generates an initial graph structure using adjacency among segments. This graph forms our initial layout for generating the virtual space layer in Metaearth architecture.

3) Semantic merging

Having constructed the virtual space layer, we can now generate the core layer and mapping layer via the semantic merging process.

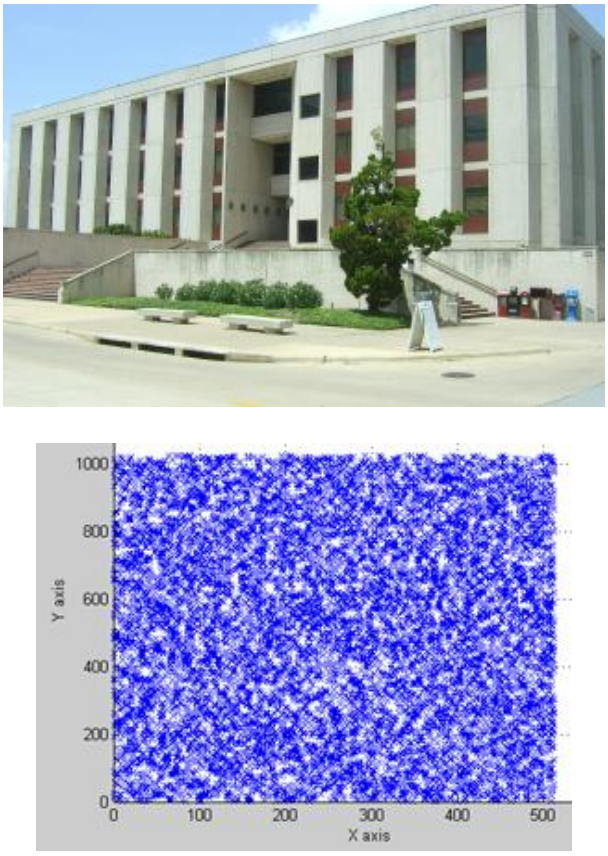
The following sections explain the three sub-processes with suitable examples and implementations.

4.2 FUZZY COLOR-BASED OVER-SEGMENTATION AND GENERATION OF VIRTUAL SPACE LAYER

4.2.1 FUZZY COLOR-BASED OVER-SEGMENTATION

Over-segmentation is considered as a preprocessing stage. Using an image with 1024×768 pixels, the initial number of overall pixels is $786{,}432 (=1024\times768)$. Note that handling around 700,000 variables is considered a large-scale optimization issue. In addition, each pixel has many properties such as true color, hue or lamination. In true color, a pixel has red (R), green (G) and blue (B). Each R, G, B value has one value out

of 0 to 255 ($=2^8$). If a pixel has a true color, the size of the variable increases to $786{,}432 \times 2^{24}$. Obviously, the increased size contributes to computation complexity; thus, in order to reduce the variables we conduct random sampling to select a certain amount of pixels. Considering the input image's height and width, we extract $1/700$ pixels. Figure 19 shows the randomly sampled pixels.



(a) A 1024×768 image (b) Randomly sampled pixels

Figure 19. Original 2D image and extracted pixels.

The true colors in the extracted pixels are selected by 1/700 scales of our original image size. Figure 20 shows the R, G, B plotting from the extracted pixels.

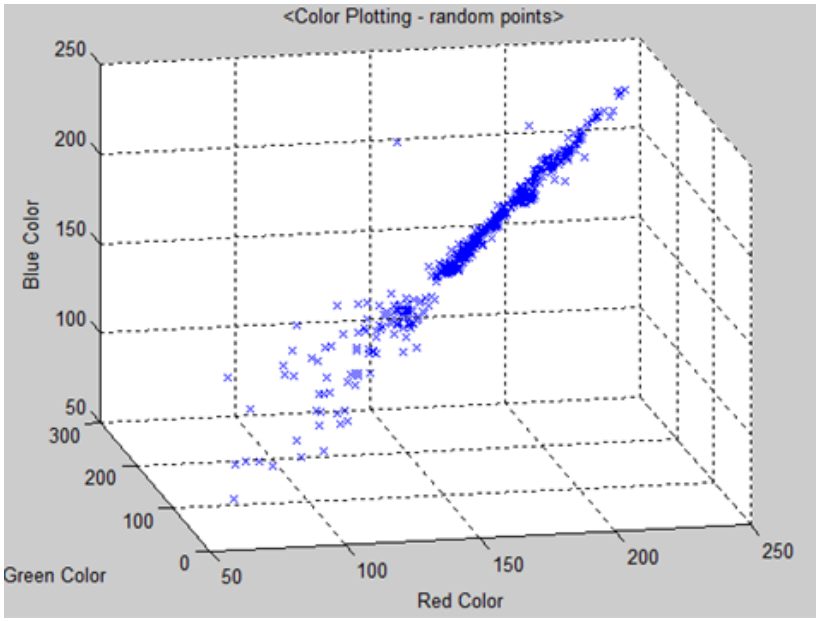


Figure 20. Red, Green and Blue color plotting from randomly extracted pixels.

The objective of this process is to obtain K-means fuzzy color quantization from the input image. The problem is to determine the K value. In most K-means algorithms or KNN approaches, the value K is assumed or determined using learning methods. Figure 21 shows the general K-means algorithm.

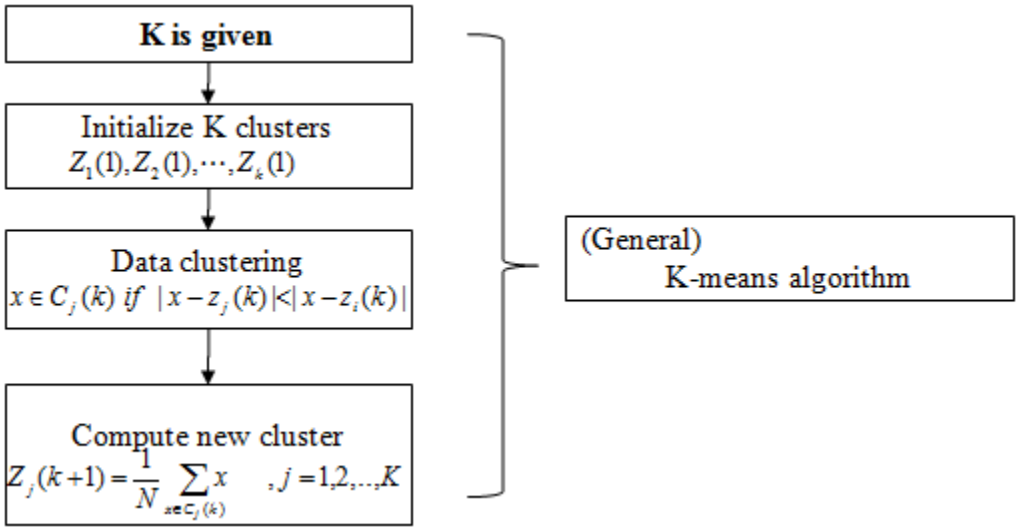


Figure 21. General procedure of K-means algorithm.

To determine an effective K value, we modify Ray and Turi's method [59]. Their method uses the intra-cluster distance measure and inner-cluster distance measure. In their method, K value is determined by some condition such as minimizing the inner-cluster scatter and maximizing the intra-cluster separation.

Definition 6. *Inner distance*

$$\textit{Inner distance} = \frac{1}{N} \sum_{i=1}^{K} \sum_{j=1} | x_{i,j} - z_i |^2$$

where, N is the number of sampled pixels

K is the number of color quantization

$i \in \{1,2,..., K\}$

$x_{i,j}$ *is* j^{th} *pixel's R,G,B vector in the* i^{th} *cluster*

and z_i *is the R,G,B center in the* i^{th} *cluster*

Definition 7. *Inter distance*

$$\textit{Inter distance} = \min_{\substack{i=1,\ldots,K-1 \\ j=i+1,\ldots,K}} (|z_i - z_j|^2)$$

where, z_i *is the R,G,B center in the* i^{th} *cluster*

and z_j *is the R,G,B center in the* j^{th} *cluster*

Using the two distance gives the ratio distance as defined:

Definition 8. *Ratio distance (K)*

$$\textit{Ratio distance (K)} = \frac{\text{Inner distance}}{\text{Inter distance}} = \frac{\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}|x_{i,j} - z_i|^2}{\min\limits_{\substack{i=1,\ldots,K-1 \\ j=i+1,\ldots,K}} (|z_i - z_j|^2)}$$

Since the desirable K can be defined by minimizing $\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}|x_{i,j} - z_i|^2$ and maximizing $\min\limits_{\substack{i=1,\ldots,K-1 \\ j=i+1,\ldots,K}} (|z_i - z_j|^2)$, the minimized ratio distance is expected to provide

the desirable K. In this manner, optimal K is calculated using a nonlinear programming in Table 5.

Table 5. K-fuzzy color determination using nonlinear programming.

- Optimal K value $= \arg\min_{K} \dfrac{\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}|x_{i,j}-z_i|^2}{\min\limits_{\substack{i=1,\ldots,K-1 \\ j=i+1,\ldots,K}}(|z_i-z_j|^2)}$
- Constraint
  - $\text{Ratio distance}(K-1) > \text{Ratio distance}(K)$
  - $\text{Ratio distance}(K) < \text{Ratio distance}(K+1)$
  - $K > 5$
  - $K \le K_{Max}$

The objective function of the nonlinear programming has four types of constraints. The first and second guarantee a locally minimizing ratio distance and the third prevents a trivial solution. Assuming K is a small number, such as $K \le 5$, an inter distance between two clusters is usually larger and can be a trivial solution. The nonlinear programming allows us to calculate an optimal K value from the input image.

Figure 22 shows this process. Thus, we obtain the value 12 as the optimal K minimizing the ratio distance.

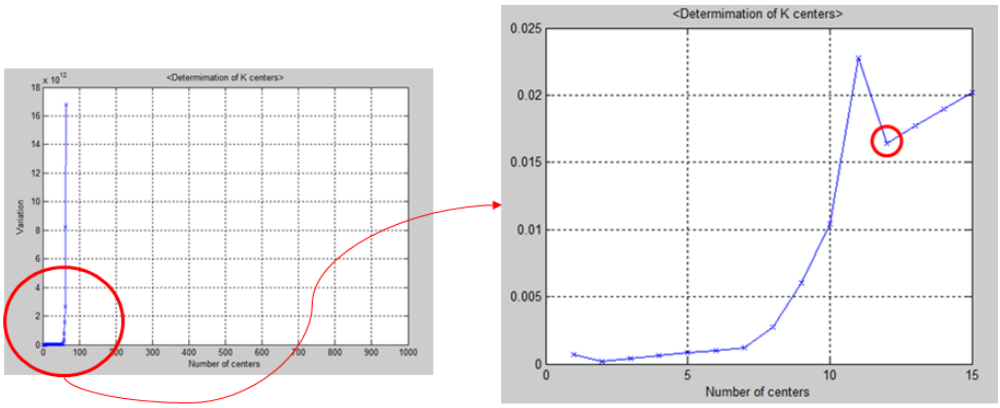


Figure 22. Determination of K using nonlinear programming.

Finally, the R, G, B color values from the randomly extracted pixels (Figure 20) are segmented into 12 regions (Figure 23).

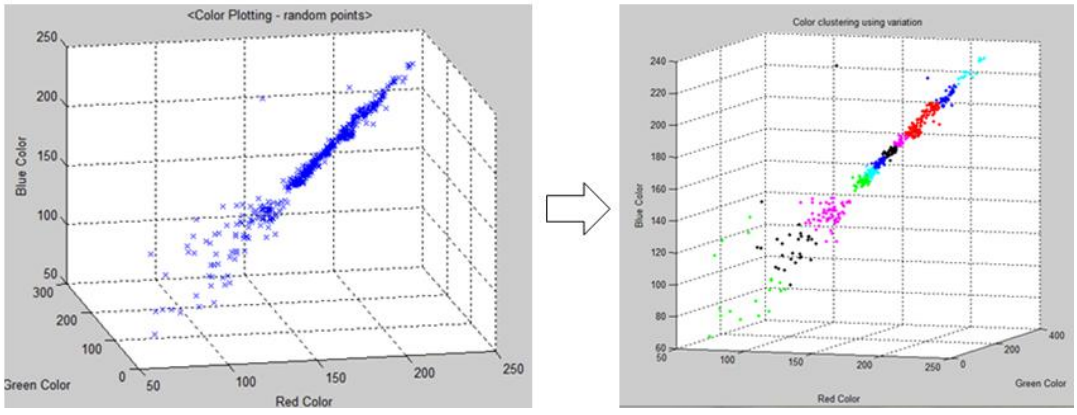


Figure 23. Segmented regions using nonlinear programming.

After segmenting color spaces are acquired using an input source, fuzzy colors are extracted from each segmented region. As this fuzzy color extraction is based on the input source, it is called "dynamic fuzzy color extraction". Other methods such as Chung and Fung [58] and Konstantinidis et al. [60] use a predefined fuzzy color or trained parameters for extracting fuzzy colors. In this concept, existing methods can be considered as "static fuzzy color generations". In this dissertation, a triangular fuzzy membership function [61] is used for defining fuzzy color. Definition 9 represents triangular based fuzzy color.

Definition 9. *Fuzzy color*

The i^{th} *segment's fuzzy color is defined by*

$$(\tilde{I}_{i,1}, \tilde{I}_{i,2}, \tilde{I}_{i,3})$$

where, $\tilde{I}_{i,1}$ *=Red fuzzy value,*

$\tilde{I}_{i,2}$ *=Green fuzzy value,*

$\tilde{I}_{i,3}$ *=Blue fuzzy value,*

Each color fuzzy value is defined by

$$\tilde{I}_{i,j;\,j=1,2,3} = \begin{cases} 0 & j^{th}\ color < \min\limits_{\forall i^{th}\ segment} j^{th}\ color \\ \dfrac{j^{th}\ color - \min\limits_{\forall i^{th}\ segment} j^{th}\ color}{\mathrm{mode}\limits_{\forall i^{th}\ segment} j^{th}\ color - \min\limits_{\forall i^{th}\ segment} j^{th}\ color} & \min\limits_{\forall i^{th}\ segment} j^{th}\ color \le j^{th}\ color \le \mathrm{mode}\limits_{\forall i^{th}\ segment} j^{th}\ color \\ \dfrac{\max\limits_{\forall i^{th}\ segment} j^{th}\ color - j^{th}\ color}{\max\limits_{\forall i^{th}\ segment} j^{th}\ color - \mathrm{mod}\limits_{\forall i^{th}\ segment} j^{th}\ color} & \mathrm{mode}\limits_{\forall i^{th}\ segment} j^{th}\ color \le j^{th}\ color \le \max\limits_{\forall i^{th}\ segment} j^{th}\ color \\ 0 & j^{th}\ color > \max\limits_{\forall i^{th}\ segment} j^{th}\ color \end{cases}$$

where,

The extraction of fuzzy color is shown in Figure 24. It offers two important advantages:

1. R,G,B true color pixels can have various $\tilde{I}_{i,j}$
2. Input image-based fuzzy color generation has superiority.

The first advantage means that a specific pixel can belong to many segments (one to many mapping) with $\tilde{I}_{i,j}$ utility values. This characteristic is useful for determining ordering the next best segment in a true color-based color.

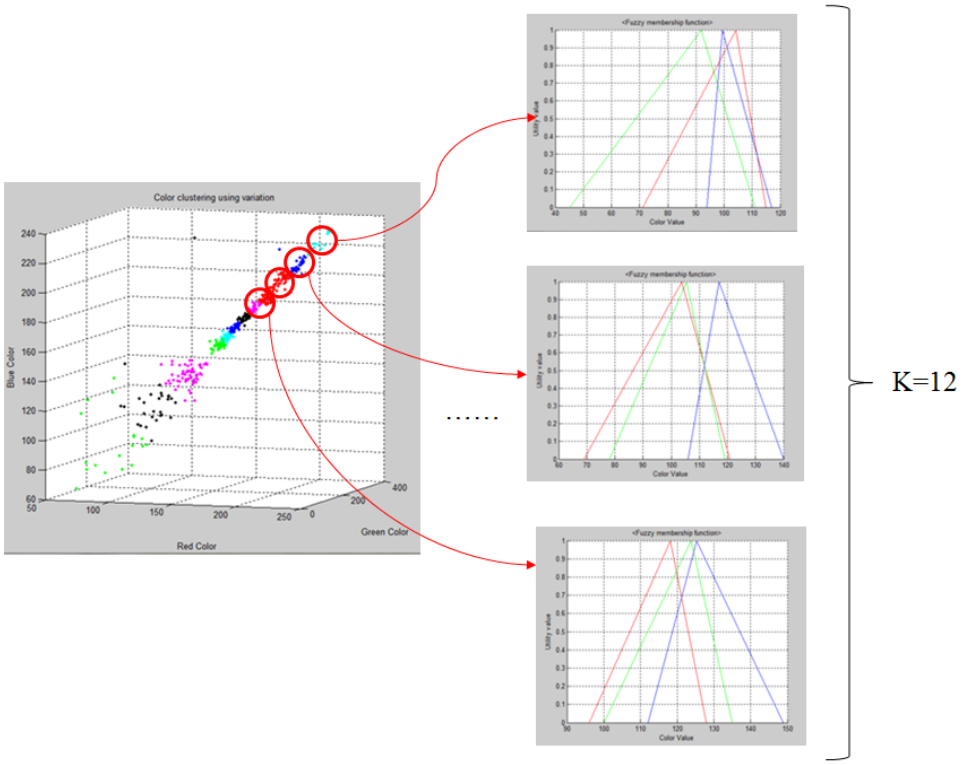


Figure 24. Fuzzy color generation.

Finally, the image is re-segmented using the generated fuzzy colors. Figure 25 shows that each red boundary-based segment is obtained by measuring the distance between a pixel's color and the generated fuzzy colors. We select the closed fuzzy color as an alternating color of the pixel. If the 8 adjacent pixels near the pixel have the same fuzzy color, it is determined as the same group.

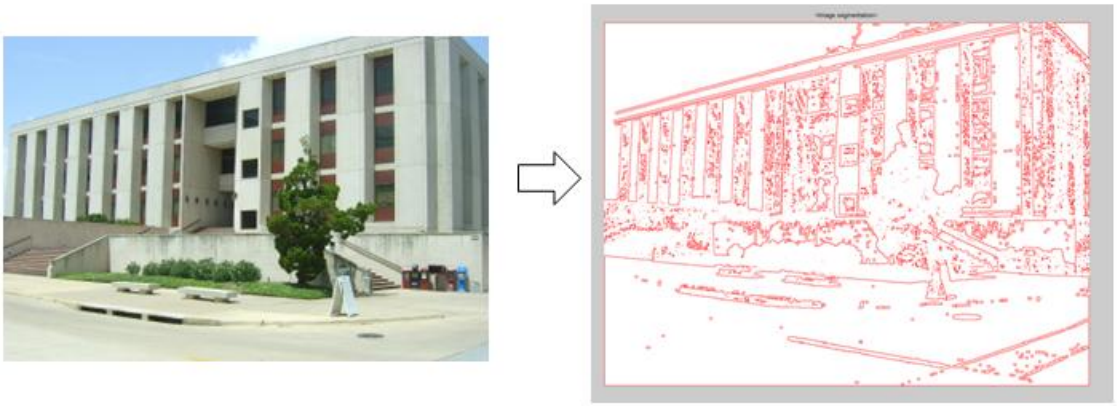
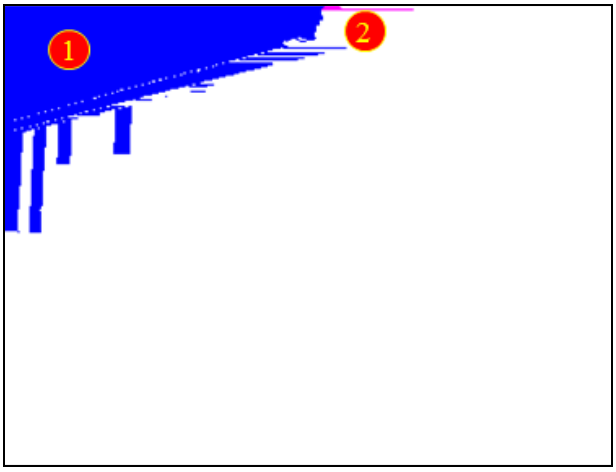


Figure 25. The initial over-segmented image.

However, fuzzy color-based segmented regions exhibit several problems, one of which is over-abstraction. Even though fuzzy color can map one pixel to multiple colors, the existence of the small fuzzy colors (= K fuzzy small colors) occurs in over-abstraction, i.e. comparatively distinct colors are grouped in one segment. Figure 26 shows an example of over-abstraction.

(a) Selected regions in original image.



(b) selected regions from the fuzzy color-based segmentation.

Figure 26. Example of over-abstraction.

From the 557 initially segmented regions, we select the first and second regions. The second region as shown in Figure 26 (b), represents part of the cloud. It can be considered a substantially over-segmented region because it has only one context

(cloud). The first region with three contexts, sky, cloud and building wall, is considered "over-abstraction". The occurrence of over-abstraction due to the small number of fuzzy colors requires re-segmenting some regions. To prevent over-abstraction, the "cell division using EM algorithm", a statistical method which is guaranteed to converge to the maximum likelihood estimator (MLE) for estimating a parameter, is proposed. An abbreviation of the Expectation-Maximization algorithm, EM algorithm is based on the concept of replacing one difficult likelihood maximization with a sequence of easier maximization whose limit is the answer to the original problem [62]. Table 6 delineates the procedure.

Table 6. Cell division process using EM algorithm.

Step 1 : In each fuzzy color segmented region, R, G, B histogram is generated.

Step 2 : Estimation of Gaussian mixture distributions

In each R,G,B histogram, a Gaussian mixture distributions is mapped.

$$f_{j;j=1,2,3} = \sum_i w_i \cdot Z(u_{i,j}, \sigma_{i,j}^2)$$

Step 2.1 : An estimation of unimodal Gaussian mixture distribution using direct fitting - regression

Step 2.2 : An estimation of multimodal Gaussian mixture distribution using EM algorithm

Table 6. Continued.

- Expectation procedures:

  calculation of $E_j[X]$ and $E_j[R \mid w_1,..w_n, \mu_1,..\mu_n, \sigma_1,..\sigma_n]$

- Maximization (Minimization) procedures
  - Objective fun. $Min \mid E_j[X] - E_j[R \mid w_1,..w_n, \mu_1,..\mu_n, \sigma_1,..\sigma_n] \mid$
  - Constraint and learning equations
    - $w_k^{new} \leftarrow w_k^{old} + \Delta w_k ; k = 1,...,n$
    - $\mu_k^{new} \leftarrow \mu_k^{old} + \Delta \mu_k ; k = 1,...,n$
    - $\sigma_k^{new} \leftarrow \sigma_k^{old} + \Delta \sigma_k ; k = 1,...,n$

Step 3 : Determination of a number of mixed Gaussian distribution in each R,G,B color

$$n_j = \arg\min_n \mid E_j[X] - \int_0^{256} R \cdot \sum_i^n w_{i,j} \cdot Z(\mu_{i,j}, \sigma_{i,j}^2) dR \mid$$

Step 4 : Cell division

$$total\ number\ of\ \ divided\ segment = \prod_{i;i=1,2,3} n_j$$

Here, we assume that each color histogram from a specific segment can be explained using Gaussian mixture distributions. Using the suggested algorithm, the over-abstracted region is divided into regions. Figure 27 shows the result.

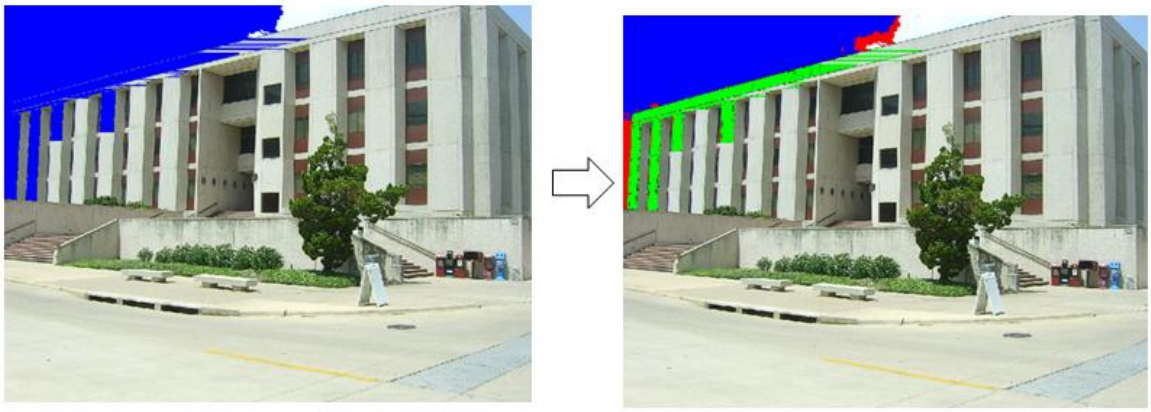


Figure 27. Result of cell division.

In this figure, one region is divided into 4 regions. The histogram of the red and blue color are fitted by a bimodal Gaussian mixed distribution and the histogram of the green color is fitted by a unimodal Gaussian mixed distribution. Finally, the 4 inner regions ($4 = 2 \cdot 1 \cdot 2$) are segmented using cell division.

Another problem with the suggested fuzzy color-based segmentation is noise. Figure 25 shows that our segmented image comprises many small areas, or “noise regions”. However, even though each has a specific color and context, we can safely ignore in the interests of over-segmentation. We use 1,000 pixels as a predefined size. Figure 28 shows the maximum size of the noise region (the blue box).

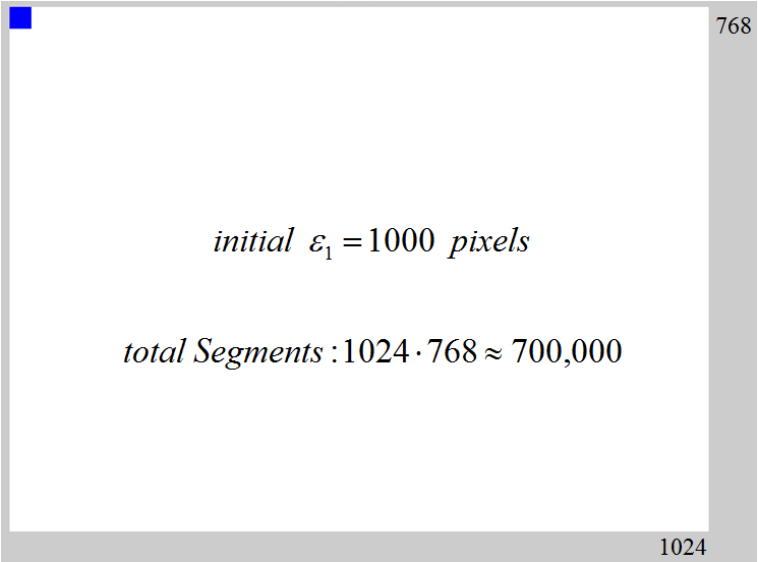


Figure 28. Pre-defined size for determining a noise region.

This noise region is merged with other noise regions or non-noise regions. As candidate cells to be merged with a noise region, adjacent regions are listed and fuzzy color intensities in each R,G,B domain are compared. If their differences are less than our pre-defined small value, we merge the noise cell with the targeted candidate cell. In this process, the threshold values are different in each R, G, B color domain. Each threshold value is determined by the estimated parameters of the Gaussian mixture distributions as shown in Table 5. From the means and variance estimator in each R,G,B domain, we calculate the range $\mu_{i,j} \pm \sigma_{i,j}$, where i represents each color domain ($1 = R,\ 2 = G$ *and* $3 = B$) and j means j^{th} as j^{th} estimator. The reason to use 1 σ is that it is enough for extracting the threshold values empirically. From each $\mu_{i,j} \pm \sigma_{i,j}$

plotting, the minimum inter-distance is calculated and used for a threshold value. Figure 29 is a graphic illustration.

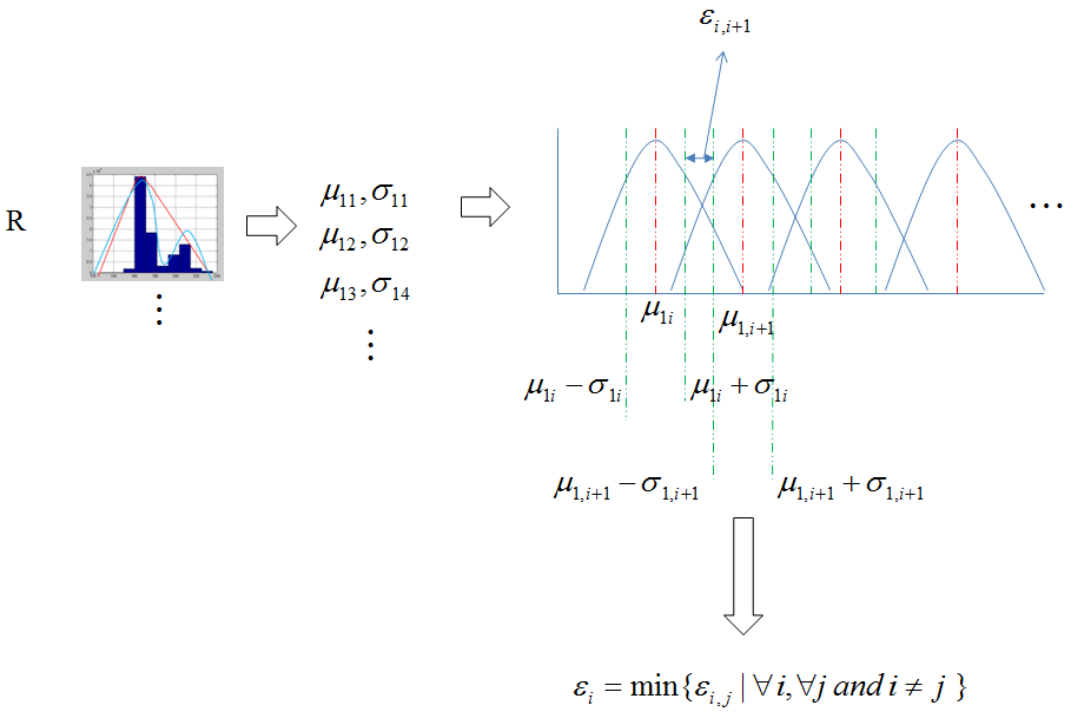


Figure 29. Extraction of a threshold value for noise region handling in the Red color space.

If the minimum value is 0 or negative value while extracting $\varepsilon_{i;i=1,2,3}$, the next positive value is determined as a $\varepsilon_{i;i=1,2,3}$. After extracting $\varepsilon_{i;i=1,2,3}$, the R,G,B intensity differences between the target noise region and adjacent regions are measured. If these intensities are less than $\varepsilon_{i;i=1,2,3}$, two regions are determined to be merged. Then we

merge the region with smaller pixels into the pixel with larger pixels. Figure 30 shows the result after cell division and noise region handling.

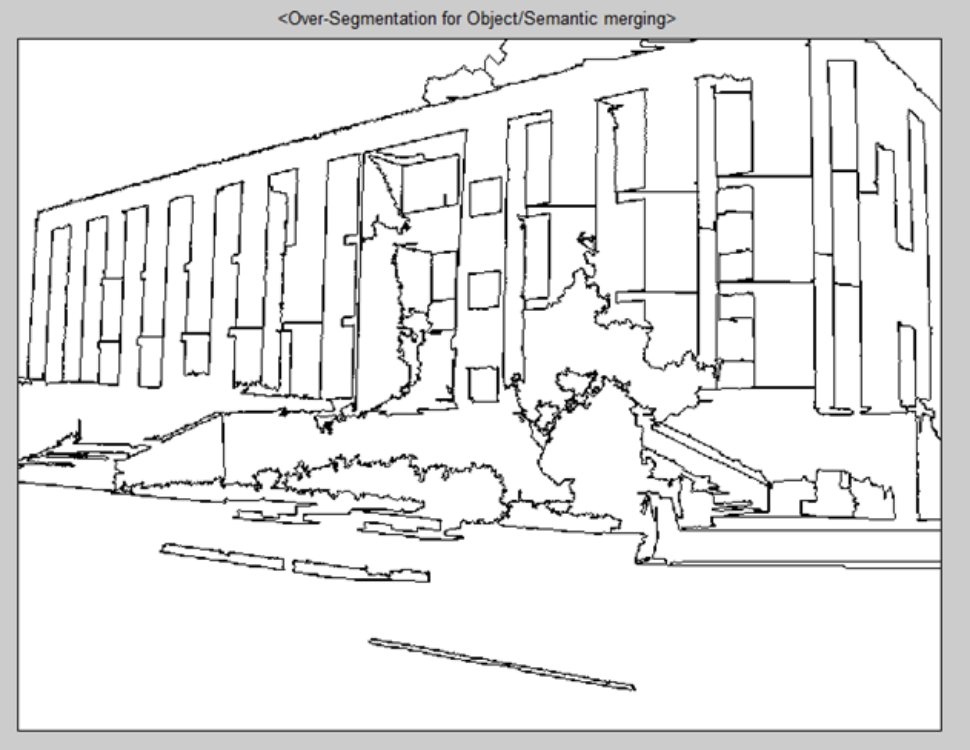


Figure 30. Result of fuzzy color-based segmentation.

Compared with the initial segmentations (Figure 25), our finalized image of 78 regions lacks noise and many over-abstracted regions.

The final step consists of constructing a virtual space layer for Metaearth architecture.

4.2.2 GENERATION OF VIRTUAL SPACE LAYER

Figure 31 shows the fuzzy color-based over-segmented region after mapping its fuzzy colors. In this example our $1024 * 768$ image is compressed into 78 regions.

Figure 31. Over-segmented region with fuzzy colors.

Our objective is the construction of Metaearth architecture containing *virtual ontology*. The following section describes the first two steps shown in Table 4: the identification of virtual components from the real world and the generation of virtual space layers. In a general 2D image, we replace the first step with the fuzzy color-based segmentation so that each region represents one virtual component. This proposition is

supported by the inference that each virtual component has unique color and the full virtual model can be assembled using these components. Thus, the over-segmented regions are converted into a graph structure using each region's bonding intensity ($BI_{i,j}$) defined as:

Definition 10. *Bonding intensity ($BI_{i,j}$)*

The bonding intensity ($BI_{i,j}$) is the number of shared pixels between i^{th} region and j^{th} region.

Proposition 1 . *The symmetry of the bonding intensity*

$$BI_{i,j} = BI_{j,i}$$

The proof of Proposition 1 is trivial. For the adjacency graph, the value in the i^{th} row and the j^{th} column is $BI_{i,j} (= BI_{j,i})$. After constructing the adjacency graph, we construct the initial graph in which each vertex represents each region. The information for a vertex and an edge appears in Figure 32.

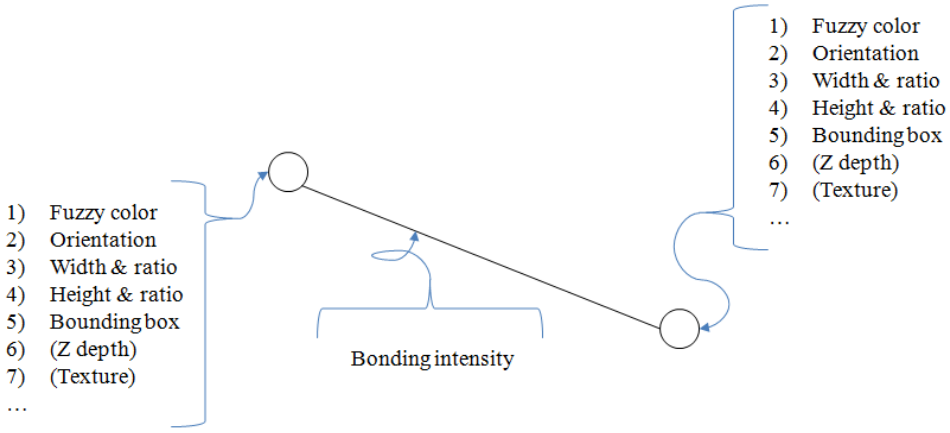


Figure 32. Elements of a vertex and an edge.

In this figure, the Z depth and texture information can be skipped by the input sources. For example, the initial graph using Section 5's methods has Z-depth information. Figure 33 shows the generated initial graph from fuzzy color-based over segmented region.

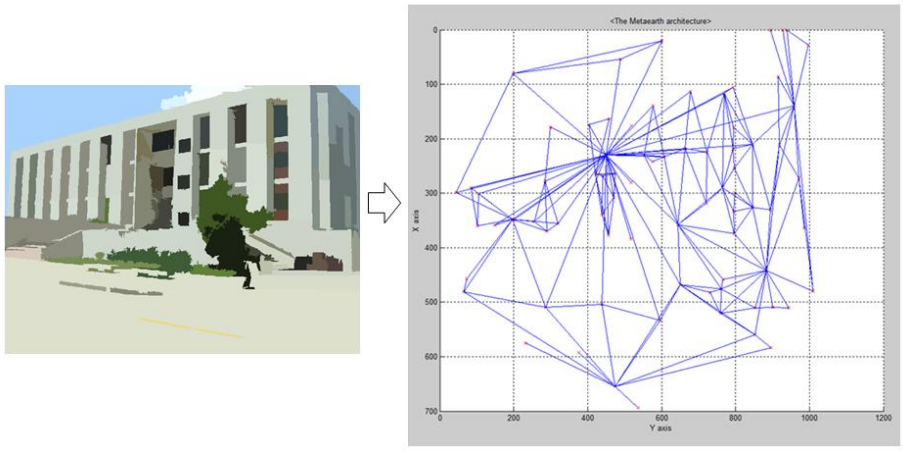


Figure 33. Conversion from fuzzy color-based segments into a graph.

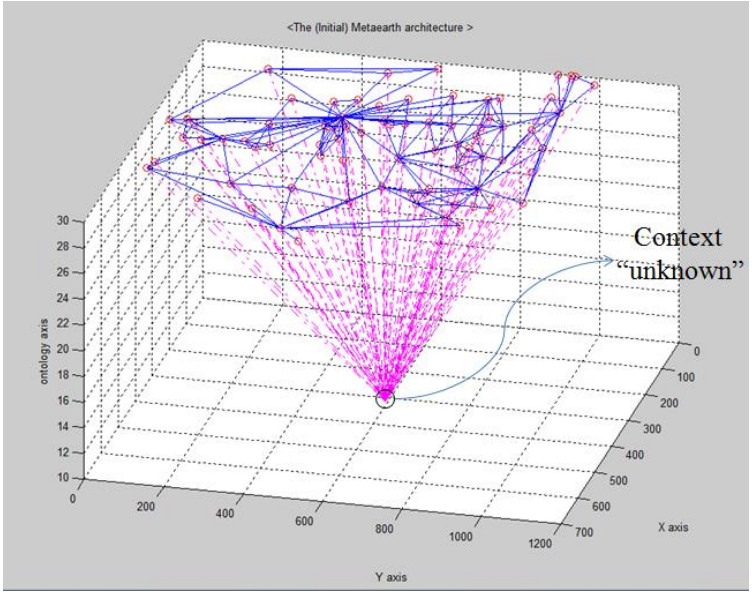


Figure 34. Initial Metaearth architecture.

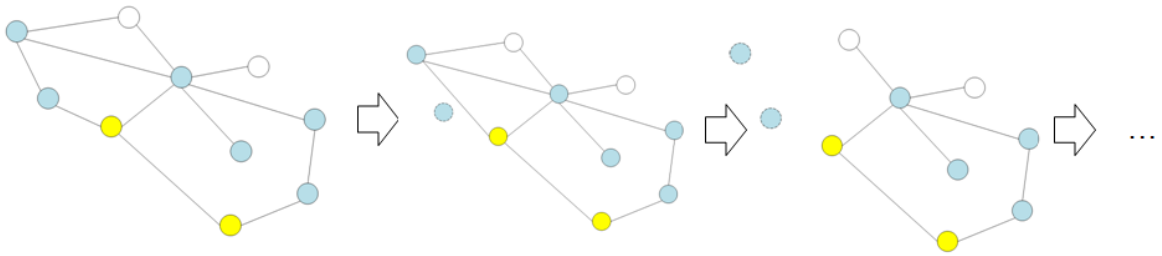
This generated graph becomes the virtual space layer in Metaearth architecture. As there is no context, each vertex is linked to a core – "Unknown". Figure 34 shows the initial Metaearth architecture. The following sections describe how the generated Metaearth architecture is intensified through the object merging process and the semantic merging process.

4. 3 OBJECT MERGING PROCESS

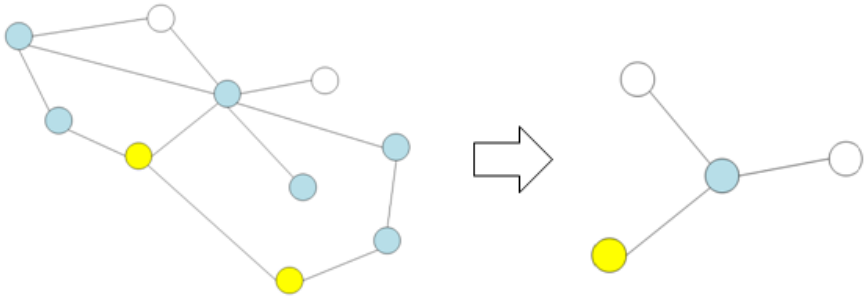
Using fuzzy color based over-segmentation, a 2D input image is segmented into over-segmented regions and the initial Metaearth architecture is constructed (Figure 34). The suggested process described in Section 4.2 is considered as the pre-processing stages for scene understanding. This section's suggested method can be considered as a main stage in scene understanding.

As discussed in section 2.2, general scene understanding techniques consist of two parts: over-segmentation and segment merging. For example, Gould *et al*. [23] use a scan line algorithm. After generating over-segmented regions, each region is compared with regions in the same horizontal line. In this comparison, pre-trained detector is used. The main limitation of their approach and similar methods is in the usage of pre-trained learning mechanisms and the output of over-segmented region. Since the output of the over-segmentation process is still segmented regions, the comparison and merging among these segments need some preprocessing such as scan line algorithm and require much time.

Since Metaearth architecture is a type of graph, the merging process is executed using graph structure in short time. Our objective is to merge similar types of regions into one in case they are connected. For example, we segment and merge three regions (blue-filled, yellow-filled and white-filled) into three; note the steps needed in the existing and suggested methods as shown in Figure 35.



(a) Segment merging process in existing techniques.



(b) Object merging process using the suggested algorithm.

Figure 35. Comparison of existing merging techniques and the suggested method.

In the suggested algorithm, this process is executed in a much shorter time. The reason is that the Meatearth architecture is a type of graph structure with adjacency matrix where each elements is *BI* as described in definition 10. In this manner, Metaearth architecture is considered a good data structure for performing object merging.

The remaining thing is to determine the similar type of regions. This determination is decided using the information presented in vertices and edges shown in figure 32. As

other scene understanding techniques do, the suggested approach compares the color, edge and shapes of objects. These information are already measured and contained in the generated initial Metaearth architecture. In this process, a region has three types of information as:

- Color : fuzzy color and true color histogram
- Edge : Bonding intensity
- Shape : Bounding box, height and width

Our remaining task is determining the regions of similar type using the information about vertices and edges as shown in Figure 32. We can then compare the color, edges and shapes of the objects.

From this data, the merging conditions are generated as:

- Color-related merging condition :

$$| \tilde{I}_{i,k} - \tilde{I}_{j,k} | < \varepsilon_{1,k;k=1,2,3} \quad \text{- (1)}$$

- Edge-related merging condition :

$$BI_{i,j} > \varepsilon_2 \quad \text{- (2)}$$

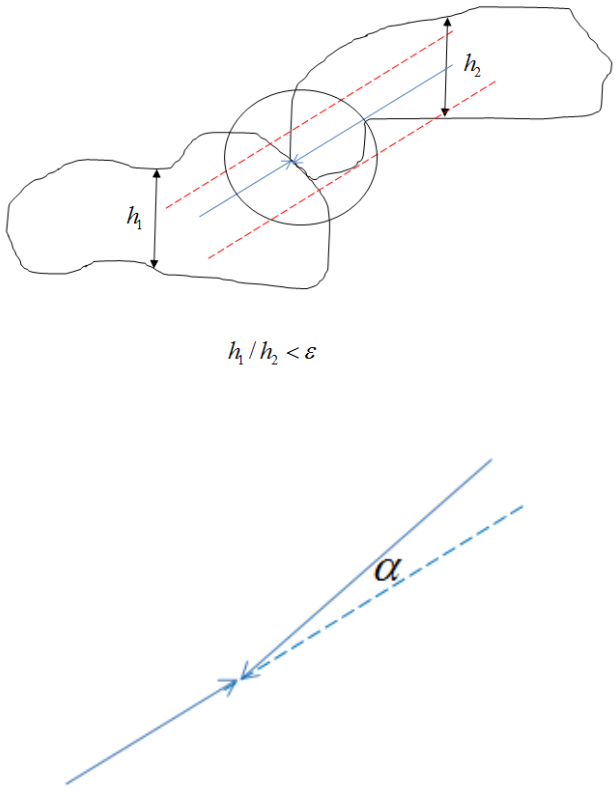
- Shape-related merging condition :

$$\varepsilon_{3,1} \le w_i / w_j \le \varepsilon_{3,2} \quad \text{- (3)}$$

$$\varepsilon_{4,1} \le h_i / h_j \le \varepsilon_{4,2} \quad \text{- (4)}$$

$$angle(l_i, l_j) \le \varepsilon_5 \quad \text{- (5)}$$

In equation (1), the difference between i^{th} region's R($k=1$), G($k=2$), B($k=3$) fuzzy color and j^{th} region's R,G,B fuzzy color intensity is compared. If this value is less than a threshold value, it meets the color-based merging condition. Similarly, edge-related condition and shape-related condition are checked using Metaearth architecture. The latter conditions, shape related condition is checked using width ratio (equation (3)), height ratio (equation (4)) and co-linearity condition (equation (5)). Figure 36 shows the height ratio test and co-linearity condition.



(a) Height ratio test (b) Co-linearity condition

Figure 36. Height ratio test and co-linearity condition for object merging.

The width, height and angle are calculated from a region's *principal component analysis* (PCA) -based axis. Finally, these equations are unified in order to determine the merging. Equation (6) shows the parameters of a unified merging equation.

$$f(\tilde{I}_{i,k;k=1,2,3}, \tilde{I}_{j,k;k=1,2,3}, BI_{i,j}, w_i, w_j, h_i, h_j, \alpha_{i,j} \mid \varepsilon_{1,k;k=1,2,3}, \varepsilon_2, \varepsilon_{3,o;o=1,2}, \varepsilon_{4,o;o=1,2}, \varepsilon_5) = \begin{cases} 0 \\ 1 \end{cases} \quad - \tag{6}$$

where, 1 means merging two regions and,

0 means the separation.

However, the shape-related merging condition does not work well in a general image. Figure 37 shows a counter-example in which two regions with quite different shapes make it difficult to apply shape-based merging conditions.

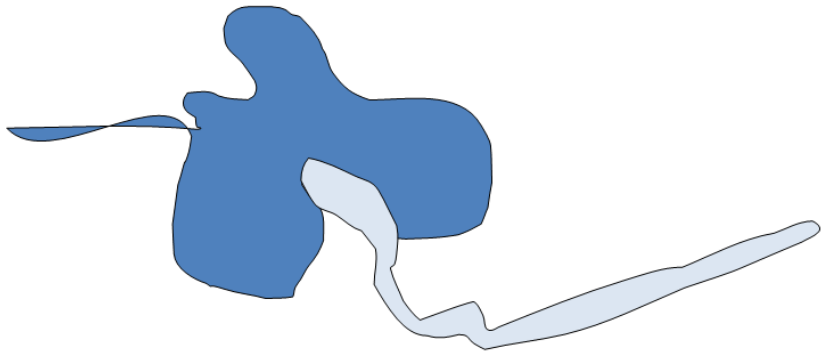


Figure 37. Difficulties of shape-related merging condition.

In general, merging conditions considering only color and edge can be used. Equation (7) shows the simplified version of equation (6).

$$f(\tilde{I}_{i,k;k=1,2,3}, \tilde{I}_{j,k;k=1,2,3}, BI_{i,j} \mid \varepsilon_{1,k;k=1,2,3}, \varepsilon_2) = \begin{cases} 0 \\ 1 \end{cases} \quad \text{- (7)}$$

where, 1 means merging two regions and,

0 means the separation.

In the particular example, the threshold values are used as $\varepsilon_{1,k;k=1,2,3} = 20$ and $\varepsilon_2 = 100$. Figure 38 shows one merging scenario using equation (6).

Finally, 78 regions are merged into 34 regions. Figure 39 shows that the vertices without any edge and the absorbed regions have been merged. The surviving regions are considered objects in Metaearth architecture. Since we have desired to decrease the vertices of the virtual space layer and to simplify it, we call the result the “intensification process of virtual space layer”.

In summary, the “intensification process of virtual space layer” shows that Metaearth architecture has been refined and prepared for the semantic-merging process.

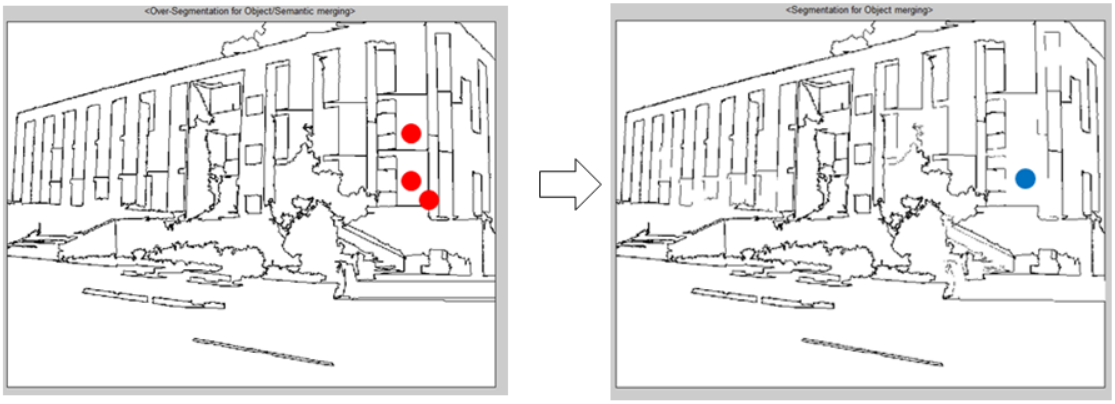


Figure 38. An example of object merging.

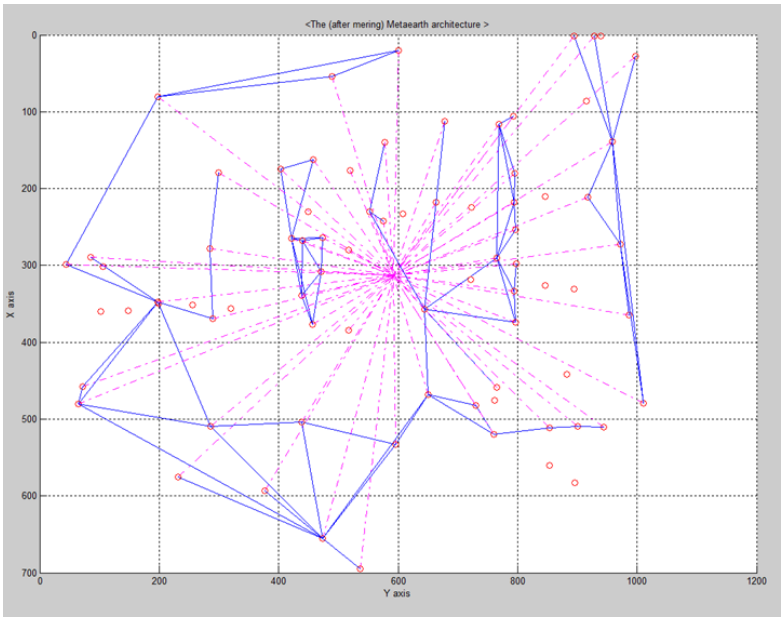


Figure 39. Metaearth architecture after object-merging process.

4. 4 SEMANTIC MERGING PROCESS

Unlike existing scene understanding techniques, the suggested method uses another merging process called the “semantic merging process”. This process is related to the generation of the ontology and mapping layers in Metaearth architecture, i.e. it represents the key process for generating *virtual ontology*. We execute it using the graph structure of Metaearth architecture.

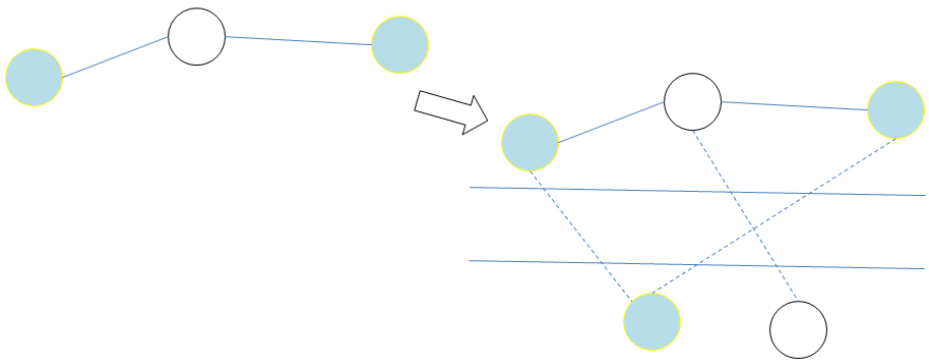


Figure 40. Semantic-merging concept.

Figure 40 shows how the semantic merging process handles non-adjacent regions. In Metaearth architecture, regions with similar fuzzy color that are not adjacent are extracted and the shape-merging conditions are checked using equations (3), (4) and (5). If these conditions are met, the regions are concluded as having the same context. Figure 41 shows an example of semantic merging. Two parts representing sky are not

adjacent. After detecting the same fuzzy color and checking the merging condition, two regions are combined into one ontology core.

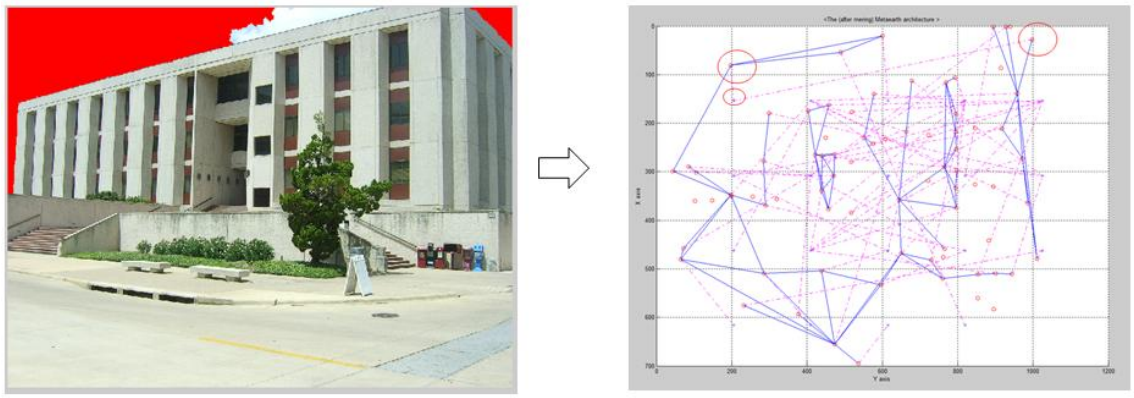


Figure 41. An example of semantic merging.

Figure 42 shows the finalized Metaearth architecture generated from the original 2D image. As a result of semantic merging, 18 ontology cores are extracted. The extracted core coincides exactly with the 18 contexts of the original image.

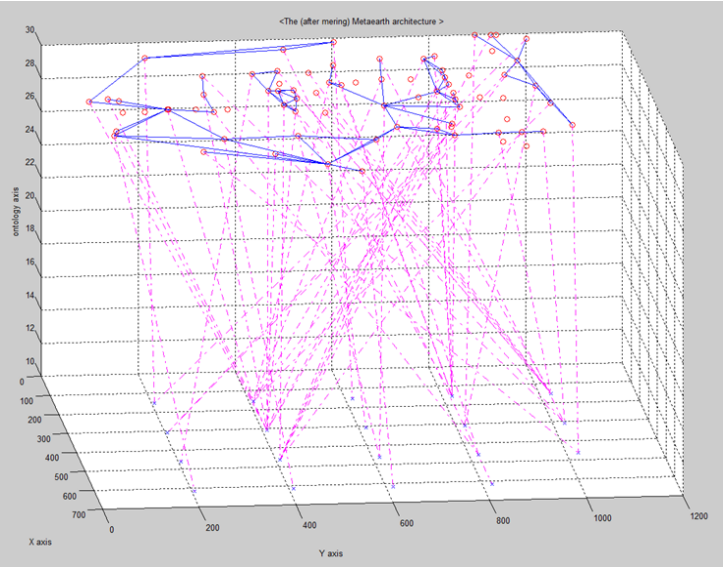


Figure 42. Metaearth architecture after semantic merging.

The next task is to map these contexts to the extracted ontology cores. The following section discusses the complexity analysis, the performance of the suggested approach and a case study.

4. 5 ALGORITHM AND SENSITIVITY ANALYSIS

The suggested approach is to extracting *virtual ontology* from a 2D scene. As the data structure, Metaearth architecture is applied. Figure 43 illustrates the overall procedure of the suggested method we now describe.

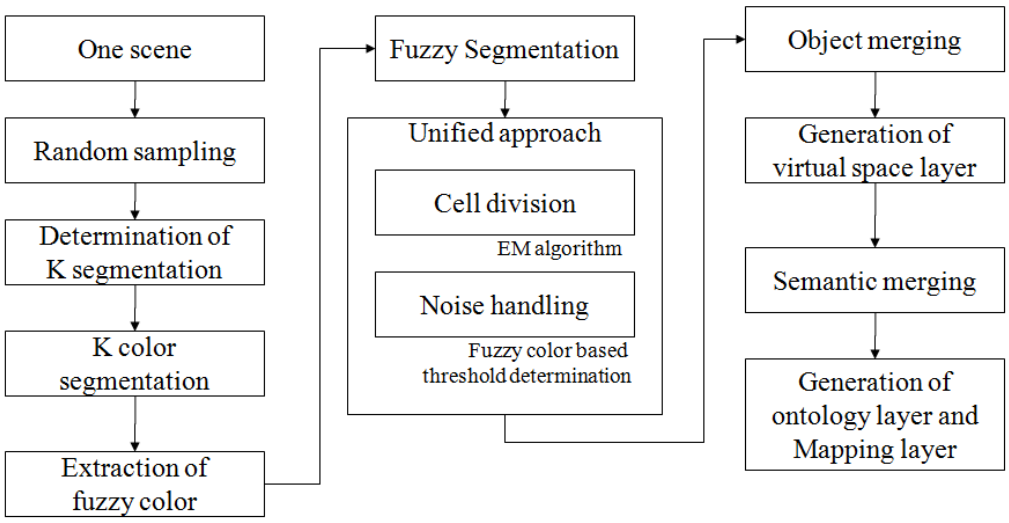


Figure 43. Overall procedure of the suggested approach.

The suggested algorithm is executed in polynomial time $O(M^2)$ where M is the number of vertices in the virtual space layer. The algorithm's complexity is summarized in Table 7.

Table 7. Complexity of the suggested algorithm.

Algorithms	Algorithm Complexity	Description
Fuzzy segmentation	$O(M^2)$	Extraction of fuzzy colors : $O(KM)$ Fuzzy segmentation : $O(M^2)$ M : the number of vertices in virtual space layer Overall algorithm : $O(M^2)=O(KM)+O(M^2)$
Object merging	$O(M^2)$	M : the number of vertices in virtual space layer
Semantic merging	$O(M^2)$	M : the number of vertices in virtual space layer
Overall process	$O(M^2)$	-

The fuzzy segmentation process is the most time-consuming of the sub-processes. While it is compared to the algorithm complexity ($\Omega(M^2)$) of Gould *et al*. [23] 's method (one of the better scene understanding algorithms), our suggested algorithm also provides good performance.

The quality of the generated Metaearth architecture depends on parameters such as the size of noise regions and several parameters of merging conditions (see Sections 4.3 and 4.4). To check the influence of each parameter, we conduct a sensitivity analysis using different combinations. We use Figure 16 (a)'s image as the input image; Table 8 compares the results with the ground truth.

Table 8. Ground truth and average color information of Figure 16(a).

No	Ground truth	Average R,G,B value		
		Red (0~255)	Green (0~255)	Blue (0~255)
1	Sky	168	208	251
2	Cloud	244	255	255
3	Wall I	206	216	207
4	Wall II	130	134	117
5	Wall III	106	71	65
6	Window I	52	69	77
7	Window II	109	118	91
8	Stair	124	109	88
9	Tree	35	48	30
10	A patch of grass I	130	163	110
11	A patch of grass II	39	70	28
12	Hole	46	48	34
13	Stand	199	212	216
14	Newspaper stand	133	59	48
15	Bench	188	197	168
16	Ground I	233	234	219
17	Ground II	191	207	207
18	Ground III	240	221	145

Table 9 shows the degree of coincidences between the ground truth and the suggested approach using different combinations of parameters. The combinations of threshold values (Figure 25) for determining noise regions and for color-based object-merging conditions (equation (1)) are used, followed by an analysis of variance (ANOVA). Figure 44 (a) show the result of two-way ANOVA test. In figure 44 (a), the column means the size-related threshold value and the row represents R/G/B color based threshold values. The result shows that two hypothesis ("color-related threshold values do not affect the performance of the suggested algorithm" and "size-related threshold values do not affect the performance of the suggested algorithm") are rejected. It means that the combination of threshold values is very crucial for obtaining well-structured virtual ontology. In order to the significance of two types of threshold values (color-related and size-related threshold values), the response surface method is applied as shown in figure 44 (b). After the test of response surface method, it shows that color-related threshold values are more significant than size-related threshold value.

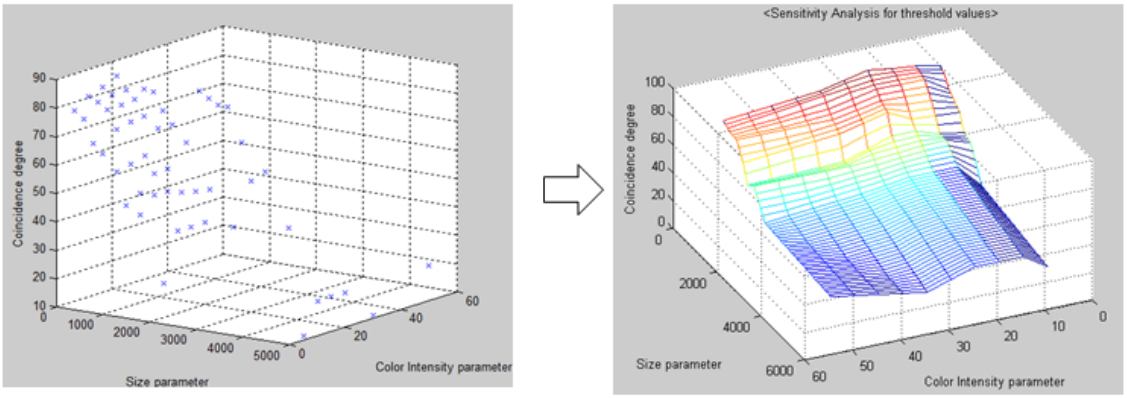
Table 9. Coincidence degree(%) between ground truth and the suggested approach using different parameters.

		COLOR INTENSITY PARAMETER																						
		Use of same color intensity value						Usage of different color intensity value																
								R	G	B	R	G	B	R	G	B	R	G	B	R	G	B	R	G	B
		5	10	15	20	30	50	10	20	10	10	10	20	20	10	10	20	10	20	20	20	10	10	20	20
SIZE	100	77.8	81.2	82.8	85.3	77.5	70.8	94.4			88.8			88.8			83.8			83.3			83.3		
	300	75.3	79.8	81.0	81.1	77.3	68.5	93.2			86.2			81.2			77.9			76.5			73.2		
	500	67.3	77.8	77.9	78.1	71.0	66.8	82.3			80.1			80.3			78.2			80.7			81.5		
	700	64.3	71.3	72.2	72.9	66.6	66.6	75.6			75.6			74.8			72.5			74.6			74.3		
	1000	58.7	59.8	61.1	69.3	61.1	55.0	69.3			67.5			67.6			61.3			64.2			61.5		
	1200	47.3	50.0	55.6	55.7	44.4	42.1	61.3			60.8			58.4			55.6			56.3			55.6		
	1500	45.0	50.3	50.0	48.3	46.0	46.1	52.3			52.1			51.3			50.7			51.9			51.2		
	2000	22.2	38.9	38.9	38.9	34.0	27.7	42.2			40.3			39.6			38.9			39.5			39.5		
	5000	11.5	22.2	22.2	22.2	22.2	11.1	22.2			22.2			22.2			22.2			22.2			22.2		

ANOVA Table

Source	SS	df	MS	F	Prob>F
Columns	803.2	5	160.64	17.88	0
Rows	20995.2	8	2624.39	292.15	0
Error	359.3	40	8.98		
Total	22157.7	53			

(a) Test result of two-way ANOVA test

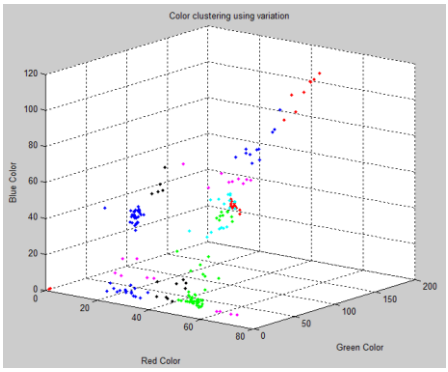


(b) the result of response surface test

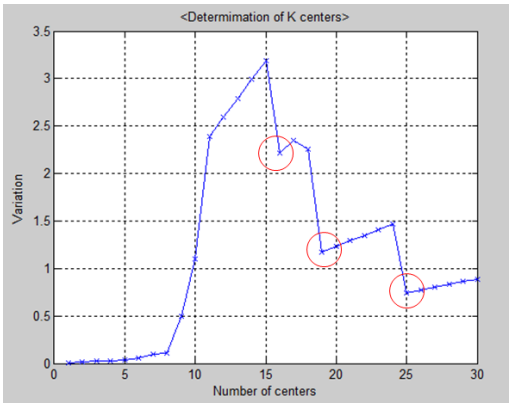
Figure 44. Sensitivity analysis using ANOVA.

With this analysis, optimal combinations can be acquired which can then be used to generate other Metaearth architecture with different input images. Figure 45 shows a Metaearth generation using a Tsukuba image. A Tsukuba image is commonly used for generating 3D disparity maps with left and right image pairs. Below, we note that the left image of the Tsukuba image shows the effectiveness of the suggested approach.

(a) Original input image

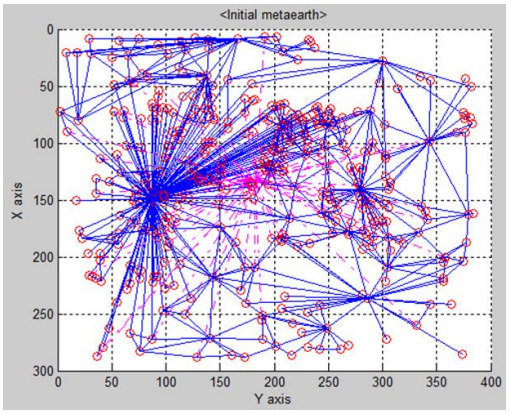


(b) Extraction of sampled pixels

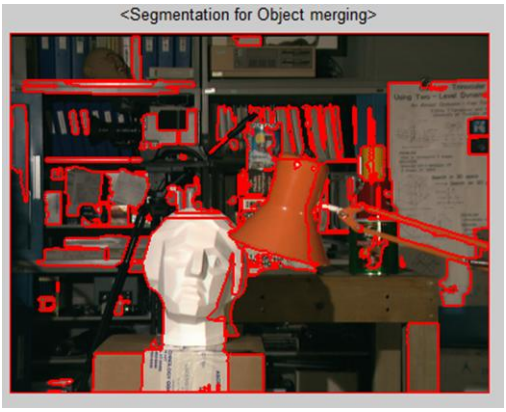


(c) Determination of K (K=16)

(d) Initial fuzzy color-based over-segmentation

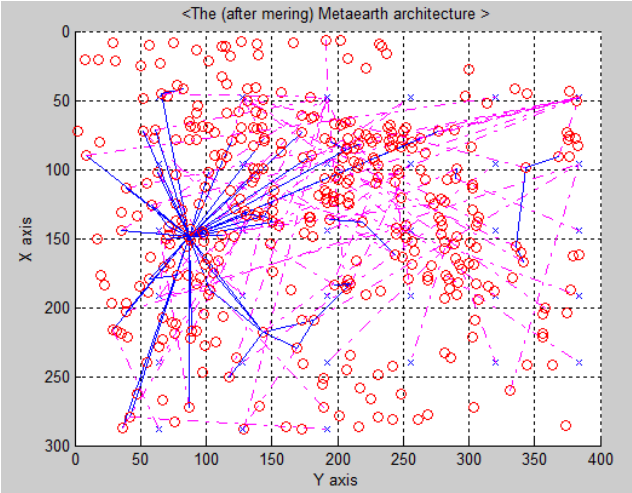


(e) Initial Metaearth architecture



(f) Image after object merging

Figure 45. Metaearth architecture from Tsukuba image.



(g) Finalized Metaearth architecture after semantic merging

Figure 45. Continued.

A limitation of the suggested method is that the generated architecture does not have Z depth because only one 2D image is used as an input. When stereo or multiview images are used, the generated architecture will have Z depth and can be used in *virtual interaction analysis*. The following section describe the generation of *virtual ontology* with Z depth.

5. CONSTRUCTION OF VIRTUAL ONTOLOGY USING MULTIVIEW SCENES

This section discusses the generation of a model with *virtual ontology.* Existing 3D reconstruction techniques and their limitations are reviewed and then a more effective method is suggested.

5. 1 ISSUES AND AMBIGUITIES IN 3D RECONSTRUCTION

In general, Z-depth information is obtained using computer vision techniques such as stereovision/multi-view approaches. The stereovision method generates a virtual model from two stereo images. Corresponding points are selected manually or automatically. Typically, the images must be adjusted to eliminate lens distortion and other forms of noise. The rectified images are then used to calculate the epipolar constraints and fundamental matrix (in cases when the camera matrix is unknown) [63]. From the fundamental matrix, two camera matrixes with intrinsic parameters and extrinsic parameters are obtained and 3D depths are generated from triangulation. Figure 46 shows the procedure. As an example, we use the general 3D computer vision technique to reconstruct the image as shown in Figure 47.

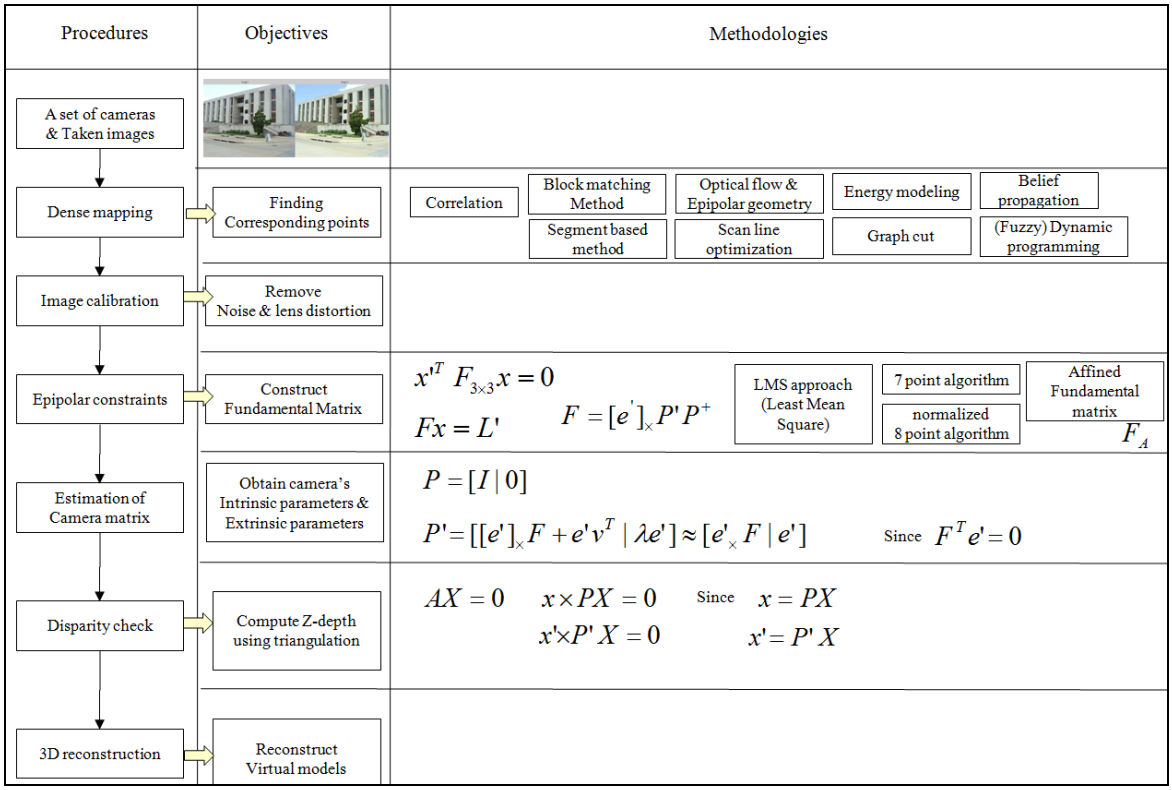
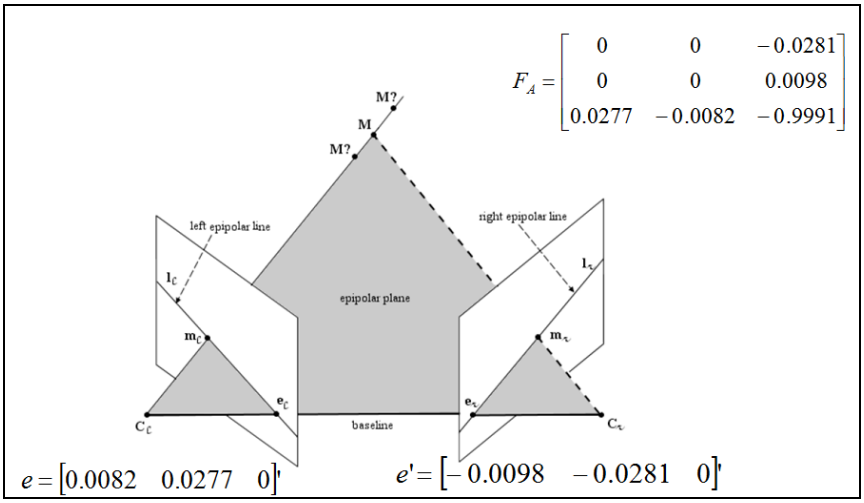


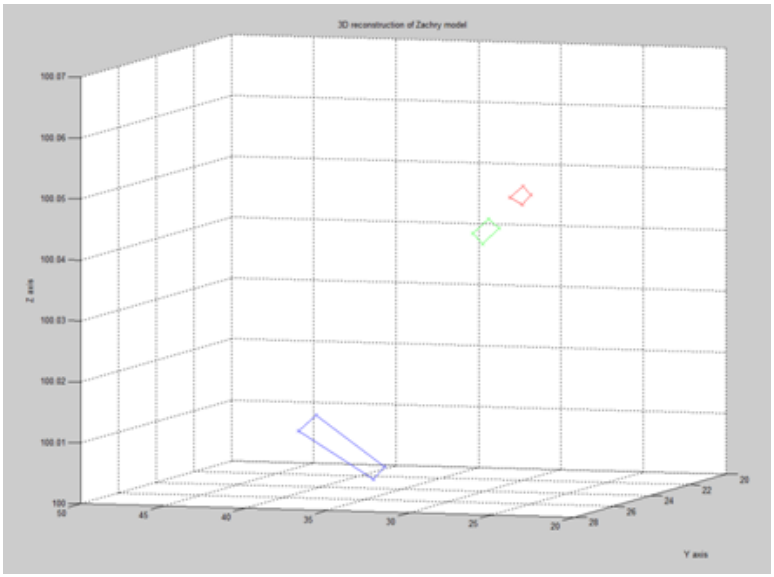
Figure 46. Stereo vision process.

(a) Left and Right image pair

Figure 47. Reconstruction process.



(b) Calculation of fundamental matrix and two epipolar constraints considering affine transformation



(c) Reconstructed regions

Figure 47. Continued.

As shown in figure 47 (c), the regions are reconstructed using RANSAC [63] image rectification and scaling up to affine transformation. However, the reconstructed regions are not satisfied due to the inaccurate reconstruction and distortions. In general, the traditional computer vision-based 3D reconstruction approaches have several problems: Table 10 summarizes the issues.

Table 10. Problems in virtual model construction.

Classification	Main issues	Problem descriptions
Construction speed	Speed and reconstruction region	- Reconstruction of full environment : Serious construction time - Reconstruction of interest regions : How to select interest region
Selection of corresponding regions	Pattern classification issue	- General issues in pattern classification : Color/geometric/ hybrid matching : Strength / limitation of pattern classification techniques
Model correction	Camera calibration issues	- Lens distortion : How to correct lens distortion - radial distortion / general distortion : How is the fixed nonlinear function exact? - Exact camera calibration
	Correction of transformation	- Which transformations can be considered up to? : Projective / affine/ similarity / Isometric transformation - How to correct 2D images as input and 3D model? : Samson error, iterative golden algorithm [63] and so on.
	Occlusion issues	- Detection of occlusions - Recovery of occluded / dis-occluded regions
Data format	Format of virtual model	Which data model can be useful for next process (simulation and analysis) : Mesh model , Voxel, VRML, Java3D, OBJ format, B-rep model, other custom formats.

Due to these issues, ambiguities are encountered in the 3D reconstruction process. There are several ambiguities which general stereovision / multiview vision methods have:

- Ambiguities in distorted image : lens distortions and various image transformation
- Ambiguity in color matching : difficulties in the selection of exact epipolar lines
- Ambiguity in occlusion handling

Figure 48 gives examples of the ambiguities in 3D reconstruction. Figure 48 (a) and (b) show an example of “ambiguity in distorted image”. Due to the camera lens’s radial and axial distortion, image pairs are distorted and two epipolar constraints with poor quality are generated. In Figure 48 (c) and (d), the brightness of the images differ on the left and right. If the reconstruction algorithm is based on color-based correspondence matching, it may fail to find accurate corresponding points.

(a) Image #1 taken from active vision

(b) Image #2 taken from active vision

(c) Left stereo image

(d) Right stereo image

(e) Left stereo image

(f) Right stereo image

Figure 48. Stereo images with occluded/non-occluded region.

Occlusion is another problem. Occluded regions can decrease the performance of correspondence matching methods such that the reconstructed regions cannot be identified as an original shape. Even though there are many occlusion-handling algorithms such as dynamic programming [64], existing approaches do not work efficiently due to many ambiguities. The right-hand side of Figure 48 (e) clearly shows that the vehicles are occluded.

Given these ambiguities, traditional computer vision techniques have one major disadvantage – the generation of an *atomic model*. Due to the absence of *virtual ontology*, the generated virtual model can be used only for visualization.

For this reason, an effective 3D reconstruction technique must have the following characteristics:

- Fast and full-scale 3D reconstruction method
- Accurate ambiguity handling
- Generation of *virtual ontology*

The following section describes a new 3D reconstruction method and how the *virtual ontology* is generated.

5. 2 FUZZY DYNAMIC PROGRAMMING AND AMBIGUITY HANDLING

To solve for ambiguities, we apply fuzzy dynamic programming (FDP) to extract Z-depth information about image pairs which we then map to Metaearth architecture. FDP was originally proposed by Jadeh and Bellman and it has been used to solve optimization problems with multi-stages and ambiguities [65, 66].

The suggested approach is termed the "hybrid *virtual ontology* generation" method. Figure 49 shows the general procedure. Multi-view images are used as the inputs and rectified to eliminate image distortion. Next, the fuzzy color-based segmentation method (Section 4) is applied. The result is that each input image now has fuzzy color and segment. FDP is applied for handling color-related ambiguities and occlusion-related ambiguities, the more accurate epipolar constraints are generated and the disparity map is calculated. Finally, Z-depth is calculated and mapped into each region generated via the fuzzy color-based segmentation method.

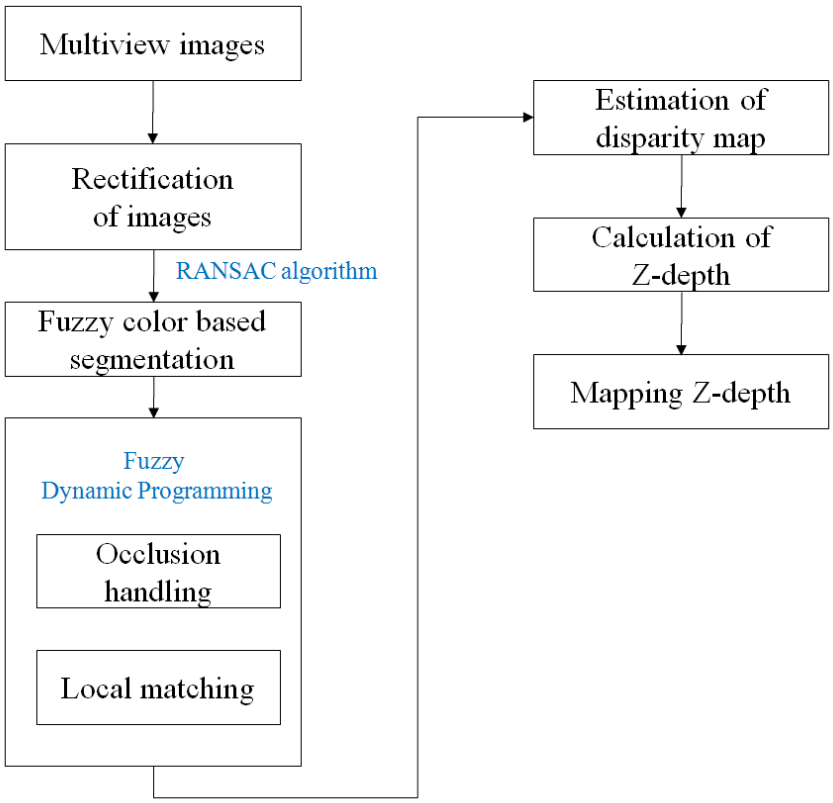


Figure 49. Procedure of Z-depth extraction using fuzzy dynamic programming.

The suggested algorithm can be compared with Klaus *et al.*'s segment-based stereo matching method [67]. The main assumption of the segment-based algorithm is that the scene can be approximated by a set of non-overlapping planes. Segment-based methods generally perform with these consecutive algorithms [67] :

1) Identify regions of homogeneous color using color segmentation methods
2) Determine disparities of reliable points using local window-based matching
3) Obtain disparity planes using the plane fitting algorithm

4) Generate an optimal disparity plane.

Even though Klaus *et al*.'s algorithm has been as a good disparity map generator experimentally, it is limited in terms of handling occlusion and is based on constant color information.

The suggested algorithm consists of five steps:

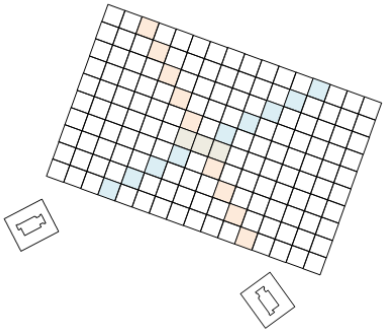
- Step 1 : Image rectification
- Step 2 : Fuzzy color-based color segmentation method
- Step 3 : Calculation of epipolar constraint using FDP
- Step 4 : Calculation of Z-depth
- Step 5 : Mapping Z-depth information

The following sections explain each step with examples.

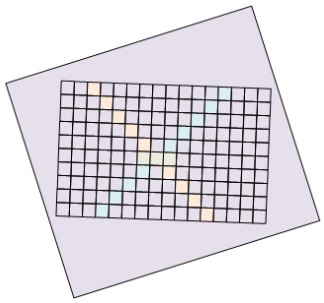
5.2.1 IMAGE RECTIFICATION AND FUZZY COLOR-BASED SEGMENTATION

The objective of image rectification is to eliminate distortions from cameras' lens and several projections such as affine transformation and projective transformation. In this dissertation, Fischler and Bolles [68] 's *RANdom Sample Consensus* (RANSAC) algorithm is applied. RANSAC method is successful and robust method in image rectification [63]. The corresponding points are matched in image pairs and 2D

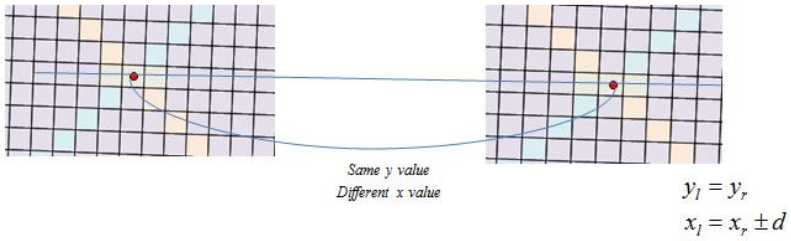
homography is computed. Using this 2D homography, the input images are rectified via the *Levenberg-Marquardt algorithm* (LM method) [63] , a nonlinear programming method. The detailed description of image rectification using RANSAC and LM method is described in Hartley and Zisserman [63]. Figure 50 shows the process.



(a) Stereo vision using two cameras



(b) Left image from stereovision system.



(c) The result of image rectification

Figure 50. Image rectification.

The image (Figure 50 (b)) from stereovision (Figure 50 (a)) may contain several distortions. Using RANSAC and LM method, each stereo image is rectified to get Figure

50 (c). As the result, the corresponding point of the targeted pixel on the epipolar line exists on the same vertical axis thus satisfying equations (8) and (9):

$$y_l = y_r \quad - \quad (8)$$

$$x_l = x_r + d' \quad - \quad (9)$$

In these equation, (x_l, y_l) and (x_r, y_r) represent the corresponding point pair in the left and right images. However, it is very difficult to satisfy equation (8), even with the application of the RANSAC estimator and LM algorithm. In real application, equation (8) is replaced by equation (10):

$$y_l = y_r + d'' \quad - \quad (10)$$

The existence of d' and d'' makes it difficult to extract exact Z-depth information. This issue is resolved in Section 5.2.2.

After rectifying the image, fuzzy color-based over-segmentation method is applied to each input image. Through this process, each image is segmented and each pixel of each image has fuzzy color. The fuzzy color is used as an input to FDP.

5.2.2 AMBIGUITY HANDLING USING FUZZY DYNAMIC PROGRAMMING

Using image rectification process and fuzzy color based over-segmentation method, corresponding points in left and right image are located in similar Y position. However, as d' and d'' (in equation (9) and (10)) exist, accurate matching may not be possible. Most stereovision approaches use block matching methods for resolving this issue, but block matching methods with FDP is used. Figure 51 shows the block matching method using fuzzy colors.

Figure 51. Block matching methods using fuzzy colors.

A targeted pixel in the left image is compared with a region within a block in the right image. In this block matching, the compared measure is nothing but fuzzy color which is extracted using fuzzy color based over-segmentation. The usage of fuzzy color can contribute to mitigating color-based ambiguity. In addition, it makes it

possible to use equation (7). With the assumption that fuzzy colors within small blocks are the same, equation (11) is derived.

$$\tilde{I}_{1,k;k=1,2,3}(x_l, y_l) = \tilde{I}_{2,k;k=1,2,3}(x_r + d', y_r) \quad - \quad (11)$$

Where, $y_l = y_r$ and $x_l = x_r + d'$

This equation is used as one constraint of FDP. We modify it from Faugeras *et al.*' dynamic programming for occlusion handling [64]. Figure 52 shows how dynamic programming can detect occluded and non-occluded regions.

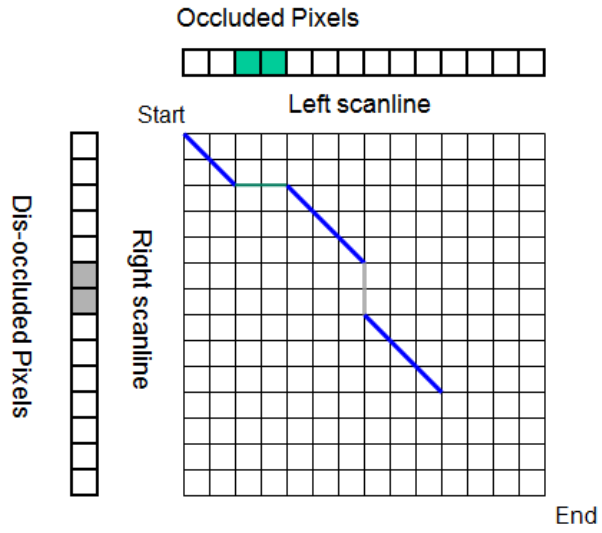


Figure 52. Faugeras *et al.*'s dynamic programming for detecting occluded regions [64].

While Faugeras *et al.*'s method has limitation in stereo image pairs with different color- based ambiguity, the suggested FDP overcomes the limitation by fuzzy colors. The objective function of FDP consists of the weighted sum of a fuzzy sum of absolute dissimilarity cost and a fuzzy gradient-based dissimilarity cost. Equation 12 defines the fuzzy sum of absolute dissimilarity cost.

$$\tilde{C}_{fuzzy\,SAD}(x,y,d)=\sum_{k=1}^{3}\sum_{(i,j)\in N(x,y)}|\tilde{I}_{1k}(i,j)-\tilde{I}_{2k}(i+d,j)| \quad - \quad (12)$$

where, $\tilde{I}_{1k}(i,j)$ is the fuzzy color intensity of i^{row} *and* j^{th} pixel in the left image, $\tilde{I}_{2k}(i+d,j)$ is the fuzzy color intensity of $(i+d)^{row}$ *and* j^{th} pixel in the right image, and N is the block size

This dissertation uses 5 as the initial block size (N), which means that a pixel in the left image can be corresponded to one pixel of 25 (5×5) pixels in the right image. The fuzzy gradient-based dissimilarity cost is defined in equation 13.

$$\tilde{C}_{fuzzy\,GRD}(x,y,d)=\sum_{k=1}^{3}\left[\sum_{(i,j)\in N_x(x,y)}|\nabla_x\tilde{I}_{1k}(i,j)-\nabla_x\tilde{I}_{2k}(i+d,j)|+\sum_{(i,j)\in N_y(x,y)}|\nabla_y\tilde{I}_{1k}(i,j)-\nabla_y\tilde{I}_{2k}(i+d,j)|\right] \quad - \quad (13)$$

where $\nabla_x\tilde{I}_{1k}$ is gradient of fuzzy color in X axis, left image

and $\nabla_y\tilde{I}_{2k}$ is gradient of fuzzy color in Y axis, right image

Figure 53 shows the conceptual example of the two costs.

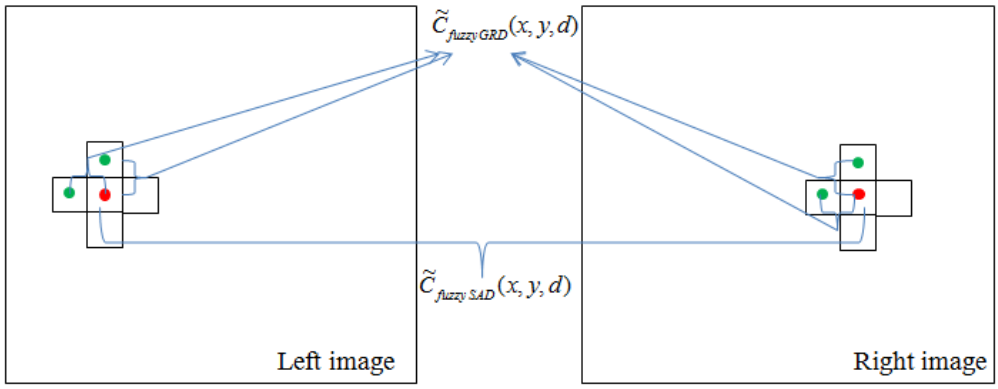


Figure 53. Fuzzy sum of absolute dissimilarity cost and fuzzy gradient-based dissimilarity cost functions.

Finally, the cost function of FDP is defined by the weighted sum of two cost functions. Equation 14 defines the cost function.

$$\tilde{C}(x,y,d) = \omega\tilde{C}_{fuzzySAD}(x,y,d) + (1-\omega)\tilde{C}_{fuzzyGRAD}(x,y,d) \quad \text{- (14)}$$

where, ω is a real number between 0 and 1

Equation (14) is the weighted cost in terms of a specific y value. A value of 0.8 is used as an initial ω. In terms of the Y axis, the total cost function of FDP is defined in Equation (15).

$$T\tilde{C}_y(x,d) = \sum_y \tilde{C}\ (x,y,d) \quad \text{- (15)}$$

The objective of FDP is to minimize $T\tilde{C}_y(x,d)$ in all y values. This objective function is combined with Faugeras *et al.*'s approach. Table 11 shows the finalized state transition equation of the proposed FDP.

Table 11. Proposed FDP method.

- Initial stage
  - $T\tilde{C}_y(1,d) = \tilde{C}(1,y,d)$
    - $Min\ \tilde{C}(1,y,d) \le d \le Max\ \tilde{C}(1,y,d)$
- State transition function
  - $T\tilde{C}_y(x,d) = T\tilde{C}_y(x-1, back\_traced) + \tilde{C}\ (x,y,d)$

The decision variable is the difference between a target pixel's x position in the left image and the X position in the right image (= d). As a constraint of d, we consider the range between the initial stage's minimum and maximum cost function values. Using the backtracking approach of dynamic programming, we obtain the minimizing difference. Finally, the accurate matching pair and occluded regions are detected. Since it generates a more accurate epipolar constraint, the exact Z depth information can be extracted. Next, we describe how Z depth information can be extracted and mapped to regions.

5.2.3 CALCULATION OF Z DEPTH AND MAPPING PROCEDURE

After detecting d using FDP, a more accurate epipolar constraint is calculated. Z-depth is calculated using these epipolar constraints and the affined fundamental matrix approach. In this dissertation, two camera matrices are assumed as affined camera matrices. Since two corresponding point pairs (x and x') satisfy equation (16), the fundamental matrix(F) is calculated using the Least Mean Square (LMS) method [69].

$$x'^{T} \cdot F \cdot x = 0 \qquad \text{- (16)}$$

From this fundamental matrix, both camera matrices (P and P') are calculated using equations (17) and (18).

$$P = [I \mid 0] \qquad \text{- (17)}$$

where I is identity matrix with 3×3,

and 0 is a zero vector with 3×1

$$P' = [e'_x F \mid e'] \qquad \text{- (18)}$$

where, e'_x is the skewed matrix [69],

F is the fundamental matrix,

and, e' is the epipolar point in the right image.

Equation (18) is derived from Equation (19) using $F^T e' = 0$.

$$P' = [e'_x F + e' v^T \mid \lambda e'] \qquad \text{- (19)}$$

Finally, global coordinate with Z depth information (X) is calculated using Equation (20).

$$A \cdot X = 0 \quad \text{- (20)}$$

Since $x = P \cdot X$ and $x' = P' \cdot X$, matrix A is derived from equation (21) and (22).

$$x \times P \cdot X = 0 \quad \text{- (21)}$$

$$x' \times P' \cdot X = 0 \quad \text{- (22)}$$

In this dissertation, the matching point x is selected from segmented regions. From a segmented region, several x are selected randomly and X is calculated using the suggested method. Finally, the region's Z depth is defined as the average Z value of the calculated X in the region. Thus, each region's pixels have the same Z depth.

5.3 GENERATION OF VIRTUAL ONTOLOGY WITH Z DEPTH INFORMATION

Z-depth information is extracted from the proposed FDP method as described in Section 5.2. The usage of fuzzy dynamic programming is caused from the extraction of fuzzy color and fuzzy color based segmentation. Then, each segmented region has the same Z depth using the method shown in section 5.2.3.

Finally, Metaearth architecture with Z depth information is generated. As described in Sections 4.2, 4.3 and 4.4, the generation sequence and procedure are the same. The difference is that a vertex in the virtual space layer has Z depth information as shown in Figure 29. This characteristic causes a great influence on object-merging process (Section 4.2) and semantic-merging process (Section 4.3). In the object-merging process, the depth-related merging condition is added.

- Z depth-related merging condition

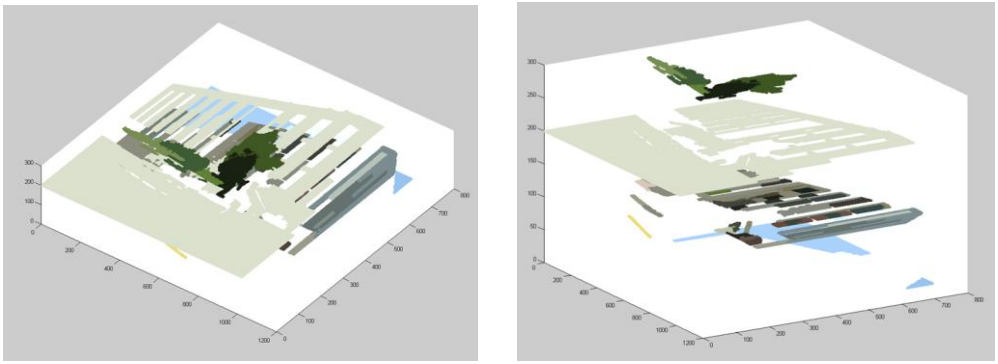
$$| Z_i - Z_j | < \varepsilon_6$$

Where, Z_i is i^{th} region's Z value.

This constraint makes it possible to distinguish objects with similar color and different distance. This constraint is added in the semantic merging condition for more clearer mapping. Figure 54 is an application of FDP.

(a) Input stereo image pairs

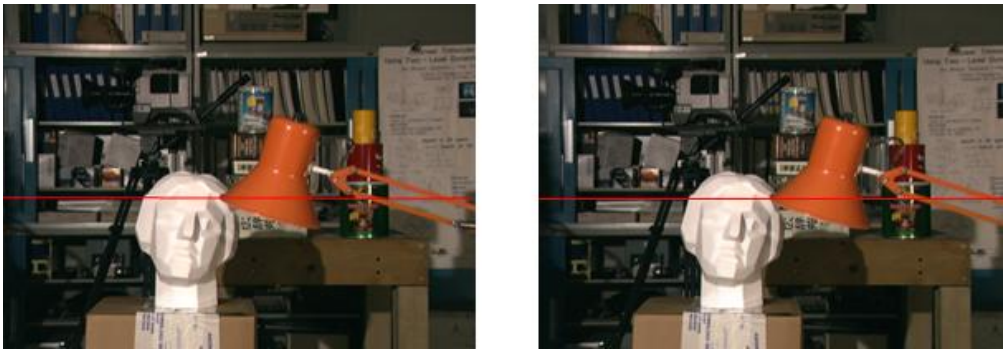
(b) Fuzzy color-based over-segmentation (from left image)



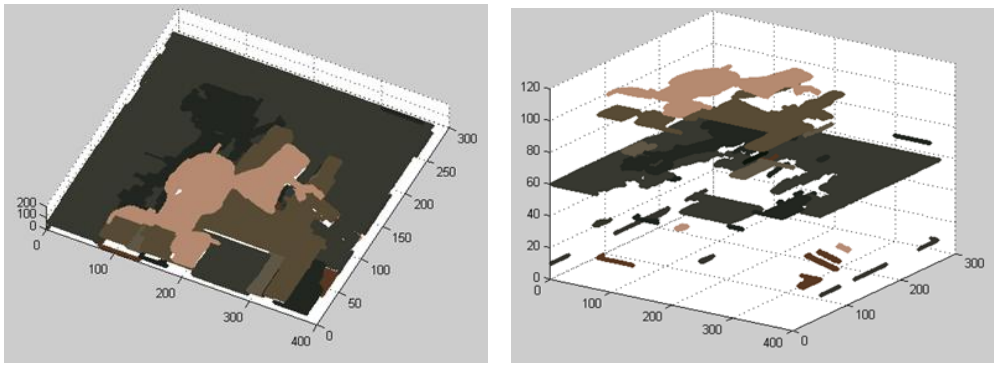
(C) Z depth mapped segmentation

Figure 54. Z depth extraction using FDP.

As another example, Tsukuba stereo image pairs are tested for generating *virtual ontology* with Z-depth. Figure 55 shows the sequence of generating *virtual ontology* using the suggested approach.

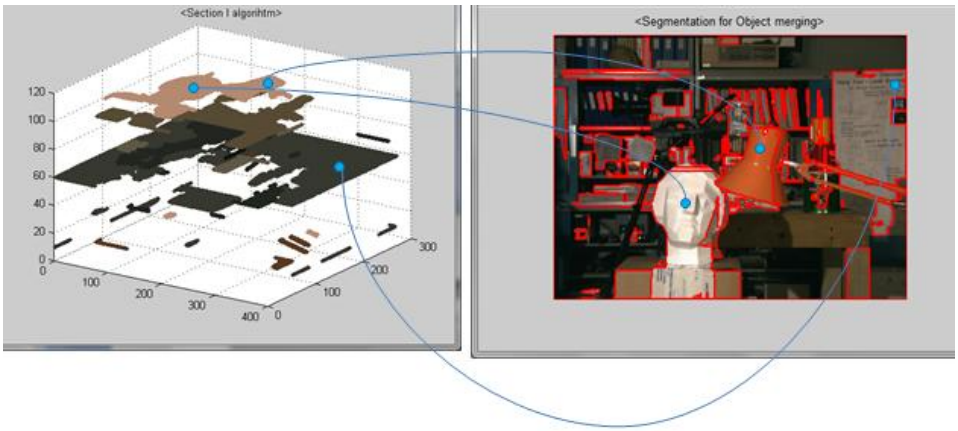


(a) Input stereo image pairs

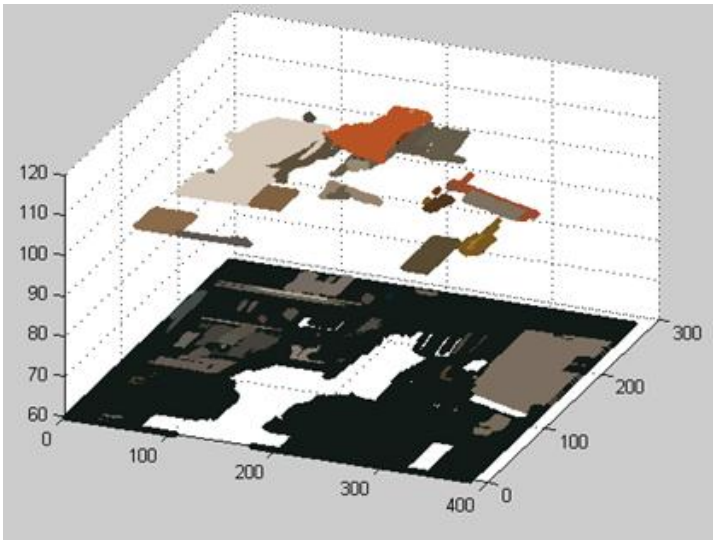


(b) Extraction of Z-depth using FDP

Figure 55. Generation of virtual ontology with Z-depth from Tsukuba stereo image pairs.

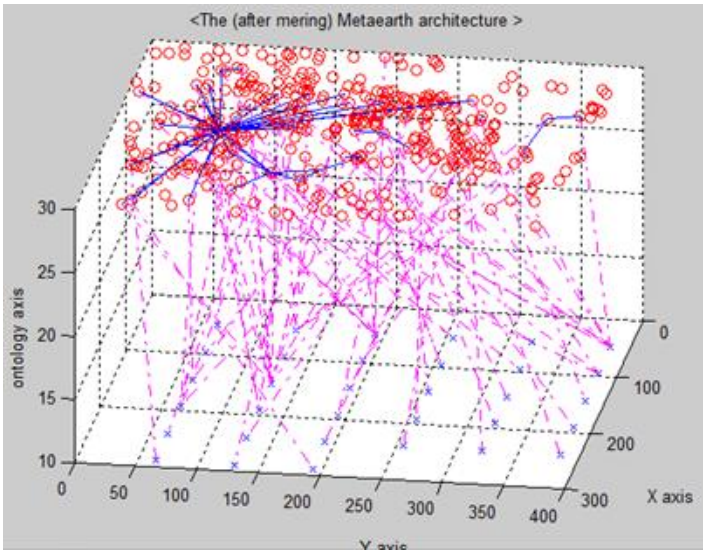


(c) Z depth mapping into fuzzy color-based segmented regions



(d) Initial virtual space layer with Z-depth

Figure 55. Continued.



(e) Finalized Metaearth architecture with Z-depth

Figure 55. Continued.

As a result of the proposed approach, the generated virtual model has *virtual ontology* with Metaearth architecture. Unlike the method in Section 4, the generated *virtual ontology* has Z depth. In other words, this model can be considered as real 3D *virtual ontology*. The remaining task is to map contexts to the generated Metaearth architecture.

6. CONTEXT MAPPING FOR VIRTUAL ONTOLOGY

This section describes how to map contexts into Metaearth architecture. Using the method presented in Sections 4 and 5, we generate the architecture from a 2D scene or multi-view scene. Recall that there are four layers (virtual space; mapping; library; ontology). We generate the virtual space layer from the fuzzy color-based segmentation process and the object-merging process. The mapping layer and the library layer are generated from the semantic-merging process. Finding that context mapping is manually conducted by operators or pre-trained detectors (whereby a poorly trained operator or detector may cause poor mapping) in similar approaches, the mapping methodology relies on graph isomorphism. Since Metaearth architecture is a type of graph as a format of *virtual ontology*, isomorphic matching properties can be applied with little error. The following sections explain graph isomorphic matching and the suggested algorithm.

6.1 GRAPH AND SUBGRAPH ISOMORPHISM

Graph isomorphism used for matching the two graphs using the two mapping functions ϕ and δ. If there are two mapping functions satisfying equations (23) and (24) in two graphs such as $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$, it is defined as G_1 is isomorphic ($G_1 \cong G_2$) to G_2 [70].

$$\varphi : V_1 \rightarrow V_2 \qquad \text{- (23)}$$

$$\text{s.t.} \qquad (\varphi(v_i), \varphi(v_j)) \in E_2$$

$$\text{for } v_i, v_j \in V_1, \ (v_i, v_j) \in E_1$$

$$\delta : V_1 \leftarrow V_2 \qquad \text{- (24)}$$

$$\text{s.t.} \qquad (\delta(v_i), \delta(v_j)) \in E_1$$

$$\text{for } v_i, v_j \in V_2, \ (v_i, v_j) \in E_2$$

This theorem can be used in many virtual reality (VR) related systems. Figure 56 shows the two scene graphs with isomorphism.

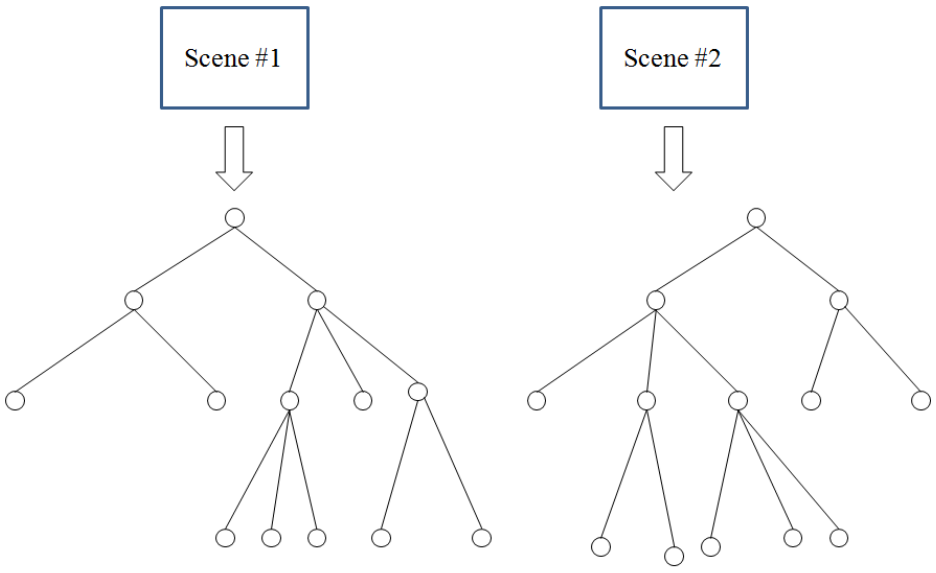


Figure 56. Isomorphism of two scene graphs.

We observe that the graphs are isomorphic using equations (22) and (23). Yet we can state only that the two scenes are identical *if* each depth's nodes are identical. In general applications, most problems belong to the category of "graph-subgraph isomorphism" which is defined as the existence of only one equation – (22) or (23).

In Figure 57 below the two graphs have a graph (G_2)-subgraph (G_1) isomorphic relationship.

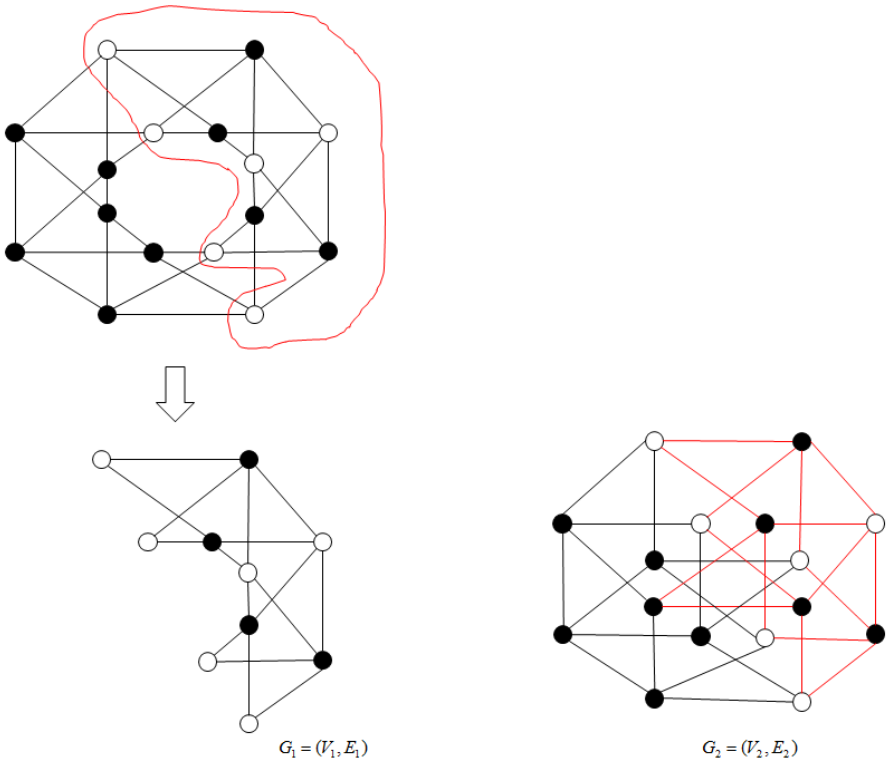


Figure 57. Graph-subgraph isomorphism between G_2 and G_1.

This concept can be used in many graph-based pattern matching problems. Figure 58 is one example of the graph-subgraph isomorphism test. The problem is

proven as NP-Complete problem by Gray [71] and a near-optimal solution is provided by researchers such as Ullmann [72] and Cordella *et al*. [73]. The comparisons among existing subgraph isomorphic matching algorithms are tested and compared in Battiti and Mascia [74] and Foggia *et al* .[75].

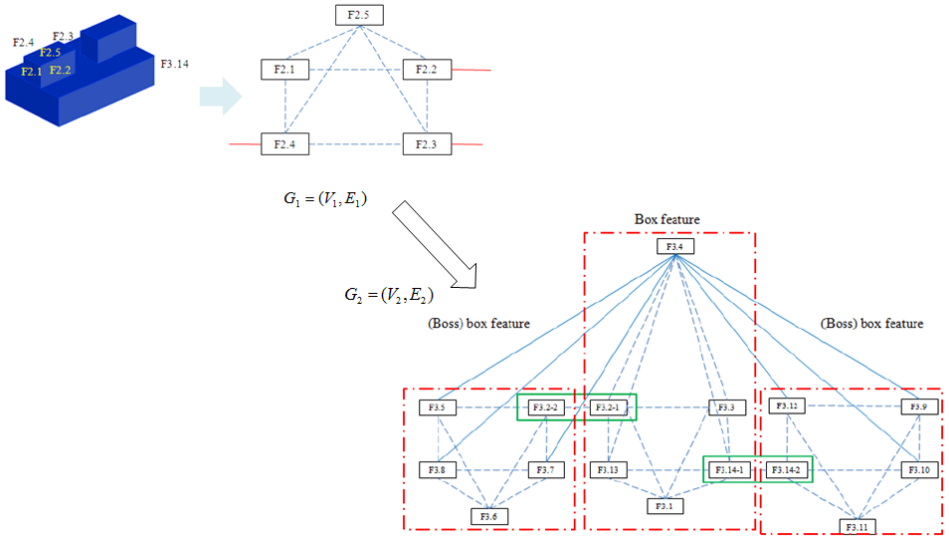
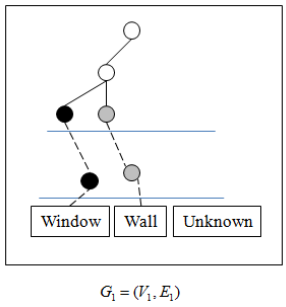


Figure 58. Graph-subgraph isomorphism in CAD feature recognition.

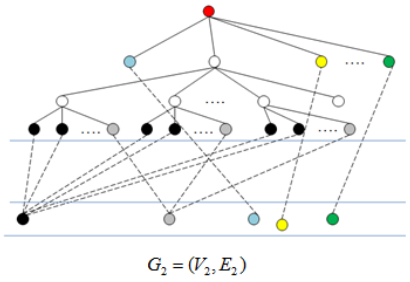
Even though it is an NP-complete problem, Messmer and Bunke [76] prove that it can be solved in polynomial time. In this dissertation, the context mapping is achieved using this graph-subgraph isomorphism. The following section describes the context mapping in Metaearth architecture.

6.2 METAEARTH-SUB METAEARTH ISOMORPHISM AND CONTEXT MAPPING

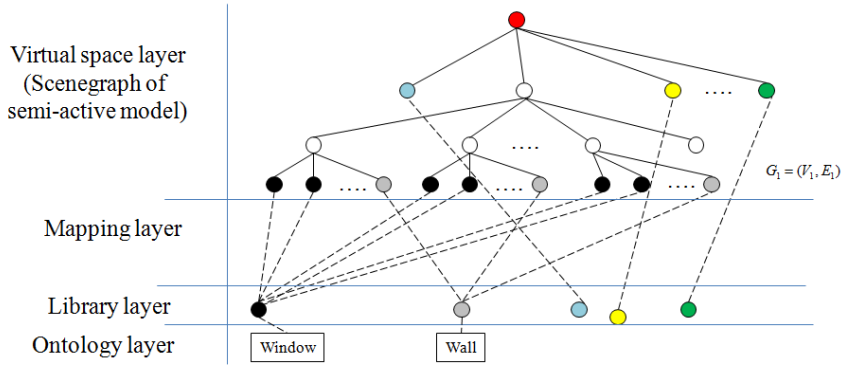
According to the section above, a subgraph can be considered as a pattern and a graph is considered as a target graph. A subgraph (G_1 in Figure 57) is used in a pattern and a graph (G_2) as a target graph with patterns which are to be detected. Our objective is to assign contexts into the ontology cores in Metaearth architecture. We use context library for the mapping. Figure 59 shows the conceptual context mapping process with context library and Metaearth architecture.



(a) Context library



(b) Metaearth architecture



(c) Metaearth architecture with contexts

Figure 59. Context mapping process with context library and Metaearth architecture.

Figure 59 (a) shows that context library has a format of Metaearth architecture, while Figure 59 (b) shows that context library's Metaearth architecture has the four complete layers. Ontology cores have contexts such as "window", "wall" or "unknown" (Figure 59 (a)) which link to each library core. These context libraries are constructed from existing Metaearth architecture or user's operation. Unlike context library, however, generated Metaearth architecture (Section 4 or 5) lacks the ontology layer. In this dissertation, context library is the pattern graph (G_2) and Metaearth architecture is the target graph (G_1). Equations (25) and (26) are the context library and the target Metaearth architecture with a representation of *Virtual ontology* as shown in Section 3.2.

$$CL := \{VC, I, C^1, MR, C^2, CR\} \quad \text{- (25)}$$

where, CL is "context library",

VC is the set of virtual components ,

I is the interactive relationship such as $I : VC \rightarrow VC$,

C^1 is the set of shared components,

MR is the mapping relationship between C^1 and VC such that $MR : C^1 \rightarrow VC$,

C^2 is the set of context,

and CR is the relationship between C^2 and C^1 such that $CR : C^2 \rightarrow C^1$

$$VO := \{VC, I, C^1, MR\} \quad \text{- (26)}$$

where, VO is "virtual ontology" without context,

VC is the set of virtual components ,

I is the interactive relationship such as $I : VC \rightarrow VC$,

C^1 is the set of shared components,

and MR is the mapping relationship between C^1 and VC such that $MR : C^1 \rightarrow VC$,

As shown in equation (26), VO does not have any context (C^2) and mapping edge (MR) to the library core. Even though CL and VO are considered as a subgraph and a graph, this problem differs from a general graph-subgraph isomorphic matching problem because:

- CL and VO have special structures such as $CL := \{VC, I, C^1, MR, C^2, CR\}$ and $VO := \{VC, I, C^1, MR\}$
- I can have a direction

A general graph-subgraph isomorphic matching problem handles a general graph such as $G = \{V, E\}$ without edge. However, Metaearth architecture has a special structure and each edge ($i \in I$) in the virtual space layer may have a direction (the interaction between two virtual components). This directed edge may be important for representing the interaction and movement of the virtual components. As an NP-complete problem (Theorem 1), it is difficult to find an exact solution.

Theorem 1. *Graph-subgraph isomorphic matching with* $CL := \{VC, I, C^1, MR, C^2, CR\}$ *and* $VO := \{VC, I, C^1, MR\}$ *is an NP-Complete problem.*

Proof. *Graph-subgraph isomorphic matching is an NP complete problem by Garey's theorem [71] which is proven using a general subgraph(*$G_1 = \{V_1, E_1\}$*) and a general graph (*$G_2 = \{V_2, E_2\}$*). For the general graph,* $CL := \{VC, I, C^1, MR, C^2, CR\}$ *graph can be converted to*

$$CL' := \{V_3, E_3\}$$

Where, $V_3 = \{VC, C^1, C^2\}$ *and* $E_3 = \{I, MR, CR\}$

and, $VO := \{VC, I, C^1, MR\}$ *can be converted to*

$$VO' := \{V_4, E_4\}$$

Where, $V_4 = \{VC, C^1\}$ *and* $E_4 = \{I, MR\}$.

Finally, $CL' := \{V_3, E_3\}$ *and* $VO' := \{V_4, E_4\}$ *are mapped into* $G_1 = \{V_1, E_1\}$ *and* $G_2 = \{V_2, E_2\}$. *Thus, it is an NP-hard problem and it can be reducible to decision problem class* C. *Ergo, it is an NP-Complete problem.*

Ullmann's algorithm [72] and Messmer and Bunke's algorithm [76] are modified in this dissertation, since it can handle a similar problem with a near-optimal solution. Their algorithms have advantages in that they can check the subgraph isomorphic matching problem and outperform in the maximal click problem. However, their algorithm is limited because it cannot find all graph-subgraph matching pairs and is only workable on general graph structure. In order to overcome the limitation of Ullmann's

algorithm and share its advantages, we modify it with Metaearth architecture. In Ullmann's algorithm and Messmer and Bunke's algorithm, graph-subgraph isomorphism is detected using permutation matrices. From G_1 and G_2, two adjacent matrices are obtained such as M_1 and M_2. Then, permutation matrix (P) is calculated from M_1 and M_2 such as each element $p_{i,j}$ of P

$$p_{i,j} \in \{0,1\}$$

$$\sum_{i=1}^{n} p_{i,j} = 1$$

$$\text{and, } \sum_{j=1}^{n} p_{i,j} = 1$$

If P exists satisfying

$$M_2 = S_{m,m}(P \cdot M_1 \cdot P^T) \quad \text{- (27)}$$

where, $S_{k,m}(A)$ is the matrix obtained from $A_{n\times n}$

by deleting rows $k+1,...,n$ and column $m+1,...,n$,

it is determined that G_2 includes G_1.

Our suggested algorithm uses and modifies their finding. Context library (CL) and the generated Metaearth architecture (VO) are the inputs. From $VO := \{VC, I, C^1, MR\}$ and $CL := \{VC, I, C^1, MR, C^2, CR\}$, two adjacency Matrices ($M_{1;m\times m}$, $M_{2;n\times n}$) are obtained considering virtual space layer (VC, I). From these

matrices, the initial permutation matrix $P_{m\times n}$ is calculated. If P does not exist, we conclude that there is no isomorphic relationship. Otherwise, all matching edges are detected using Ullmann's backtracking method. After detecting all pairs, the mapping relationship between MR_1 and MR_2 is examined. If each edge in MR_1 is mapped into MR_2, we conclude that two Metaearth architectures have the graph-subgraph isomorphic problem. Then, CL's C^2 is mapped into VO's C^1 using the mapping relation (CR_1). Figure 60 shows the suggested procedure.

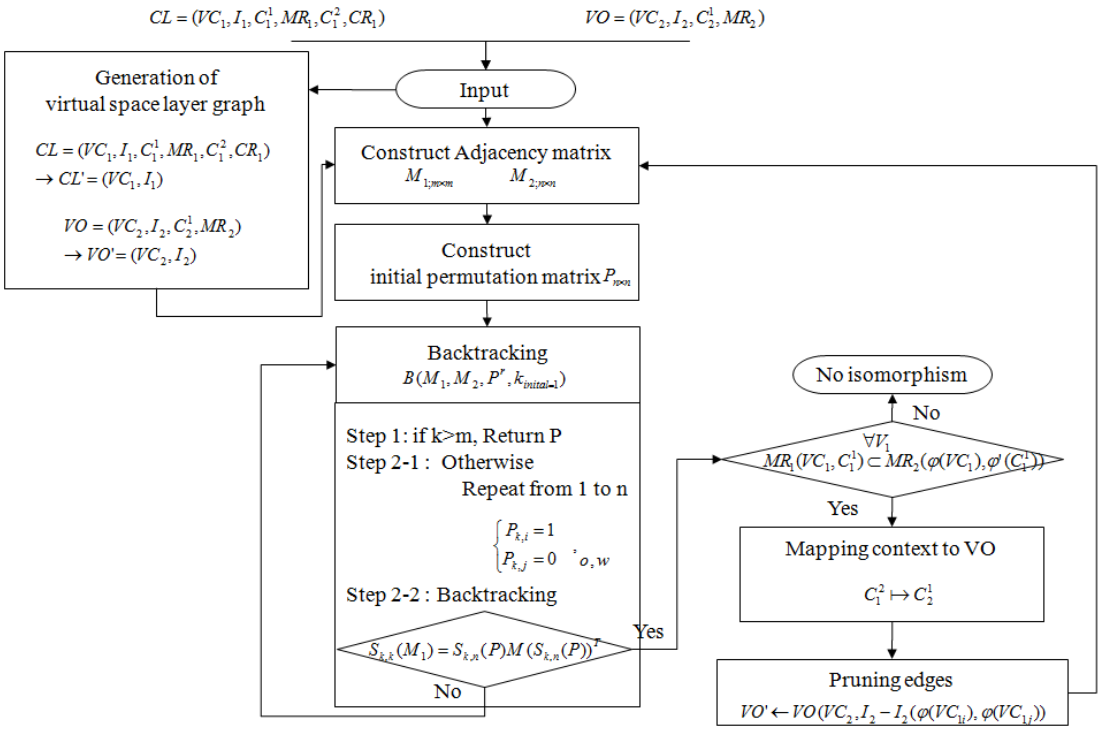


Figure 60. Context mapping in Meteaearth architecture using graph-subgraph isomorphic matching.

Our suggested algorithm can detect all subgraphs (context library). However, the size of the generated Metaearth architecture is large and performance decreases. Table 12 shows the comparison of algorithm complexity.

Table 12. Comparison of algorithm complexity between Ullmann's algorithm and the suggested algorithm.

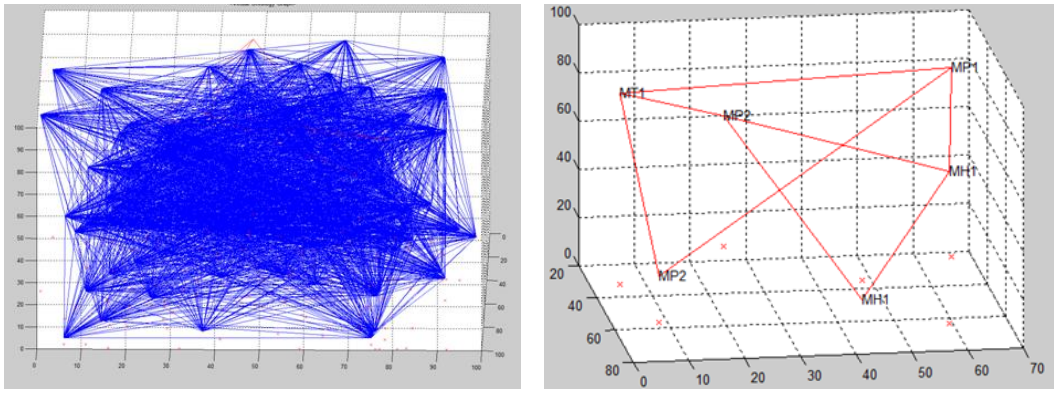
	Ullmann's method	Context mapping using Metaearth architecture
Space consumption	$O(m^2n)$	$O(m^2n)$
Best case Complexity	$O(mn)$	$O(mn)$
Worst case complexity	$O(m^2n^n)$	$O(m^2n^{n+1})$

(where, m is the size of VC_1 in context library (CL) and n is the size of VC_2 in targeted Metaearth architecture (VO))

As shown in Table 11, the complexity of the suggested algorithm is similar to Ullmann's method. Considering that Metaearth architecture has a more complicated structure than a general graph, our suggested algorithm is a workable solution. The following section describes its effectiveness using a simulation study.

6.3 SIMULATION AND ANALYSIS OF CONTEXT MAPPING METHOD

As a simulation study, a Metaearth architecture with 100 virtual components ($|VC|=100$) and 2510 interactions ($|I|=2510$) is generated.



(a) Targeted Metaearth architecture (b) Context library

Figure 61. Generated Metaearth architecture.

Figures 61 (a) and (b) show the generated Metaearth architecture and a context library with 6 virtual components, 8 interactions and 6 contexts. Each context in the context library belongs to MH1 (Material Handler #1), MP1 (Material Processor #1), MP2 (Material Processor #2) and MT1 (Material Translator #1). Using Ullmann's algorithm [72] and Messmer and Bunke's algorithm [76], 373 subgraphs are detected. Figure 62 shows parts of the detected subgraphs. However, the detected subgraphs cannot be considered as the structures with isomorphic relationship in Metaearth architecture, because, each subgraph structure may have different library cores.

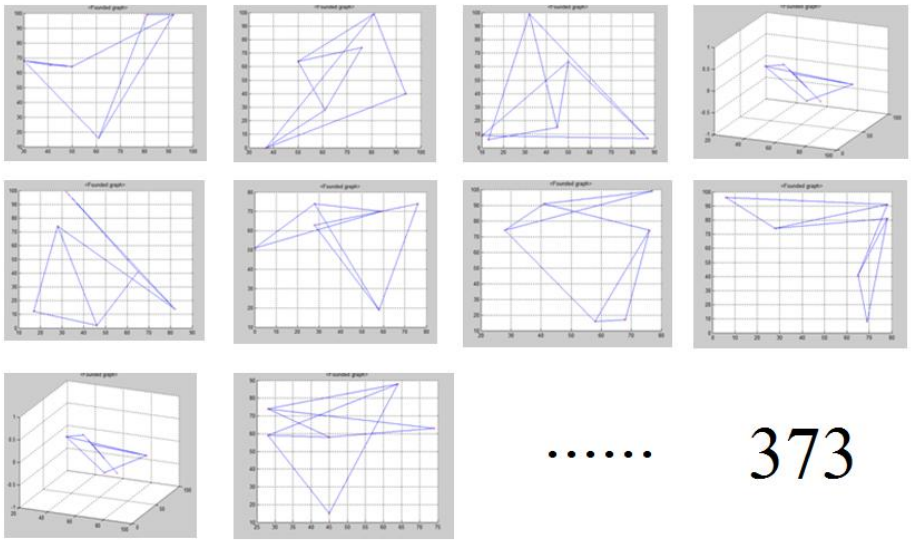
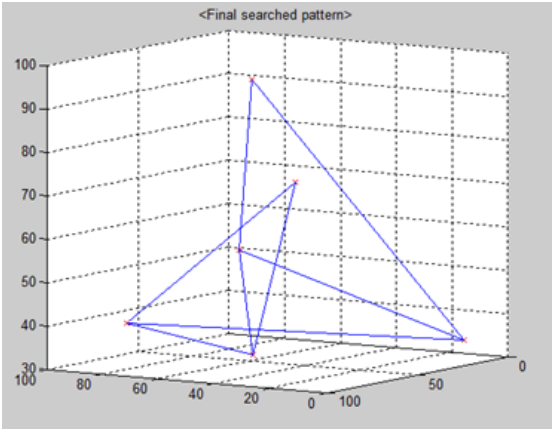


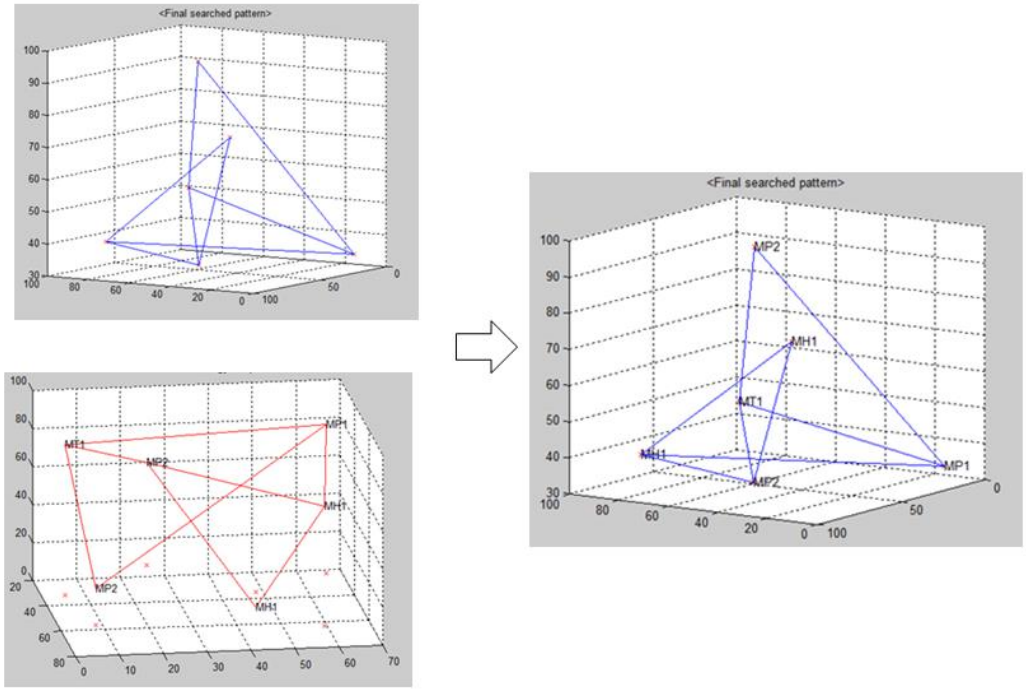
Figure 62. Parts of detected subgraph using existing algorithms.

The suggested algorithm extracts the exact pattern as shown in Figure 63 (a). Then, the contexts are mapped into the detected structure (Figure 63(b)).



(a) Detected structure with the sample structure of context library

Figure 63. Detection and context mapping.



(b) context mapping

Figure 63. Continued.

The suggested algorithm is effective in detecting the same structure with context library and mapping contexts. However, this algorithm only works with well-defined context libraries. The construction of context library and variation are discussed in the following section.

7. CONCLUSION

7. 1 RESEARCH ISSUES AND FURTHER STUDY

In this dissertation, the application domain is fixed as a generation of virtual environment. “Knowledge of virtual environment” is represented as contexts. A 2D image is considered as “unknown environment”. The structure of knowledge is represented with Metaearth architecture. *Virtual ontology* is constructed as a context mapping procedure into Metaearth architecture.

Section 3 introduces Metaearth architecture as a good format of *virtual ontology*. Since Metaearth architecture can replace other formats in terms of scene graph and support construction of large scale virtual environments (LSVE), more theoretical investigation and effort to optimize this structure are required. In addition, even though construction methods are proposed, another construction method considering various application domains is considered.

As characteristics of the suggested *virtual ontology* generation method, these do not depend heavily on prior knowledge and human operation. Considering the input image’s visual cue such as fuzzy colors, edge and shape, input images are over-segmented and merged using conditions. In this dissertation, fuzzy color and edge information are used for more important criteria than shape information such as width ratio, height ratio, and bounding box. For more clarified segmentation with good quality, shape information can be considered with higher weight. In this concept, Voronoi

diagram (VD) approach can be applied. If each segment is divided into more detail using the characteristics of Voronoi region (each Voronoi region has a convex shape), bounding box approach and width/height ratio approaches can be applied more easily to generate accurate *virtual ontology*.

Even though the suggested approaches are independent from the pre-trained mechanisms, the determination of several parameters for object merging and semantic merging are significant issues. The quality of the generated *virtual ontology* depends on these parameters. If training methods such as Neural Nets and statistical learning methods are applied to determine these parameters, more accurate virtual ontology can be generated.

Section 4's algorithm generates *virtual ontology* from a 2D image, but the generated model does not have Z depth information. If some existing reasoning methods as described in Section 2 are applied, *virtual ontology* with Z depth can be generated.
The approach in Section 5 constructs *virtual ontology* with Z-depth. It has many advantages for handling ambiguities, and offers rapid 3D reconstruction and *virtual ontology* generation. However, it can be more effective via the unifying approach with RANSAC and fuzzy color segmentation which rectify the input image pairs. If two consecutive processes are merged and unified, the algorithm complexity will be much lower and improve the speed.

Section 6 describes how the contexts are mapped to Metaearth architecture. An approach using graph-subgraph isomorphism is proposed and tested. The prior condition of this approach is in the prior construction of context libraries which are another type of

Metaearth architecture. In a real situation, the construction of context library can be considered a problem. Even though the suggested methods have a target for generating *virtual ontology* automatically, the generation and preparation of context library needs human's intervention such as mapping contexts to the library cores. The accurate context mapping can guarantee more accurate generation of *virtual ontology*. As a further study, the context library can be generated from existing Metaearth architecture. Using graph cut and detection methods, a Metaearth architecture can be divided into many small architectures and each divulged structure can be used as a context library with contexts. Finally, the generated virtual environment with *virtual ontology* is used in *virtual interaction analysis* as a next stage. Since *virtual interaction analysis* requires mutual interactions among virtual components, seamless processing from input to *virtual interaction analysis* can be achieved using the suggested method.

7. 2 SUMMARY AND CONTRIBUTIONS

This dissertation proposes how *virtual ontology* can be constructed from an unknown environment. Virtual ontology is defined as a structure of virtual environment with semantics. In this dissertation, the contextual knowledge is used as a representation of semantics. While many existing 3D reconstruction approaches generate *atomic virtual*

model without *virtual ontology*, this research suggests the generation of *relational models* and *models with virtual ontology*.

Metaearth architecture is introduced as the data structure of *virtual ontology.* The architecture consists of a virtual space layer, mapping layer, library layer and ontology layers. In the virtual space layer, the interaction and relationship between virtual components can be described. Each virtual component in the virtual space layer is connected to library cores in the library layer, thus contributing to the design of LSVEs with less redundancy. The library cores are then linked to the ontology core, a contextual knowledge.

Section 4 generates a *relational model* from a 2D input image. Unlike other scene understanding techniques, the suggested method extracts *virtual ontology* using Metaearth architecture. As an intermediate process, fuzzy color based over-segmentation method makes it possible to generate scene segmentation without pre-defined knowledge or human intervention. In addition, the scene structure is generated through the semantic-merging process.

While Section 4's model lacks Z-depth information, Section 5's model generates *virtual ontology* with Z-depth information. To handle many ambiguities in extracting 3D information, fuzzy dynamic programming is applied. A hybrid approach with FDP and Section 4's algorithm generates more accurate *virtual ontology* with Z-depth.

Finally, the generated *relational model* is converted into a *model with structural knowledge* using a context mapping process. Contextual knowledge is mapped into Metaearth architecture of the *model with structural knowledge* via the suggested

isomorphic matching method. The *relational model*'s Metaearth architecture is effectively compared to the context library's Metaearth architecture and the contexts in context library are mapped into the targeted Metaearth architecture.

The overall procedure contributes to the generation of *models with structural knowledge* from 2D images. The successful generation of *models with structural knowledge* can guarantee automatic and autonomous processing in *virtual interaction analysis*. This approach can generate an LSVE with *virtual ontology* and intelligence in less time and with less computational effort. It can also increase the use of generated virtual models and enhance seamless information processing.

REFERENCES

1. Gardner, L., *Automated Manufacturing*. 1985: ASTM STP 862.

2. Jaynes, C., Seales, W. B., Calvert, K, Fei, Z. and Griffioen, J., *The Metaverse - a networked collection of inexpensive self-configuring, immersive environments.* Eurographics workshop on Virtual Environments, 2003: p. 115-123.

3. Stepenson, N., *Snow crash*. 1993: United States, Spectra Books.

4. Lamotte, W., P. Quax, and E. Flerackers, *Large-scale networked virtual environment: architecture and applications.* Campus-Wide Information Systems, 2008. **25**(5): p. 329-341.

5. Keller, J. and G. Simon, *Solipsis: a massively multi-participant virtual world.*: p. 1-5, International conference on Parallel and Distributed Techniques and Applications, 2003.

6. Hariri, B., S. Shirmohammadi, and M.R. Pakravan, *A distributed interest management scheme for massively multi-user virtual environments.* Virtual Environment, Human-Computer Interfaces and Measurement Systems, 2008: p1-5.

7. Morse, K.L., L. Bic, and M. Dillencourt, *Interest management in large-scale virtual environment.* Presence: Teleoperators and Virtual Environments, 2000. **9**: p. 52-88.

8. Lee, H. and A. Banerjee, *A self-configurable large-scale virtual manufacturing environment for collaborative designers.* to appear in Virtual Reality, accepted 12/20099 (DOI:10.1007/s10055-009-0151-0), 2009.

9. Eppstein, D., *Subgraph Isomorphism in Planar Graphs and Related Problems.* Journal of Graph Algorithm and Applications, 1999. **3**(3): p. 1-27.

10. Cornelis, N., Leibe, B., Cornelis, K., Gool, L. V., Leuven, K. and Zurich, E., *3D urban scene modeling integrating recognition and reconstruction.* International Journal of Computer Vision, 2007. **78**(2-3): p. 121-141.

11. Grimsdale, R.L. and Lambourn, *Generation of virtual reality environments using expert systems.* International Conferences in Central Europe on Computer Graphics, Visualization and Computer Vision, 1997: p. 153-162.

12. Grimsdale, R.L. and S.M. Lambourn, *Generation of virtual reality environments using expert systems.*: International Conferences in Central Europe on Computer Graphcis, Visualization and Computer Vision, 1997: p. 153-162.

13. Lechner, T., Watson, B. and Wilensky. U., *Procedural city modeling.* 1st Midwestern Graphics Conference, 2003: p. 1-6.

14. Biggers, K., J. Keyser, and J. Wall, *Automated reconstruction of synthesized environments from complex point cloud datasets.* Interservice/Industry Training Simulation and Education Conference, 2009.

15. Tse, R.O.C., C.M. Gold, and D.B. Kindner, *Building reconstruction using LIDAR data.* International Society of Photogrammetry and Remote Sensing Workshop on updating Geo-spatial databases with imagery & the 5th International Society of Photogrammetry and Remote Sensing Workshop on DMGISs, 2004: p. 121-127.

16. Gianfranco, F., Nardinocchi, C., Scaioni, M. and Zingaretti, G., *Complete classification of raw LIDAR data and 3D reconstruction of building.* Pattern Analysis and Applications, 2006. **8**: p. 357-374.

17. Kada, M. and L. McKinley, *3D building reconstruction from LIDAR based on a cell decomposition approach.* International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2009. **3**: p. 47-53.

18. Baltsavias, E.P., *A comparison between phtogrammetry and laser scanning.* Journal of Photogrammetry & Remote Sensing, 1999. **54**: p. 83-94.

19. Baltsavias, E.P., *Airbone laser scanning : basic relations and formula.* Journal of Photogrammetry & Remote Sensing, 1999. **54**(199-214).

20. Prusinkiewicz, P., Muendermann, L., Karwowski, R. and Lane, B., *The use of positional information in the modeling of plants.* International Conference on Computer Graphics and Interactive Techniques, 2001: p. 289-300.

21. Saxena, A., M. Sun, and A.Y. Ng, *Make3D : learning 3D scene structure from a single still image.* IEEE Transactions of Pattern Analysis and Machine Intelligence, 2009. **30**(5): p. 824-840.

22. Murphy, K., A. Torralba, and W. Freeman., *Using the forest to see the trees: a graphical model relating features, objects and scenes.* Neural Information Processing Systems, 2004: p. 289-300.

23. Gould, S., R. Fulton, and D. Koller, *Decomposing a scene into geometric and semantically consistent regions.* Proceedings of the IEEE International Conference on Computer Vision, 2009: p. 1-8.

24. Hoiem, D., A. Efros, and M. Hebert, *Geometric context from a single image.* International Conference on Computer Vision, 2005: p 1-8.

25. Tu, Z., Chen, X., Yuille A. L., and Zhu S., *Image parsing: unifying segmentation, detection and recognition.* Proceedings of the Ninth International Conference on Computer Vision, 2003. **2**: p. 1-8.

26. He, X., R. Zemel, and M. Carreira-Perpinan, *Multiscale CRFs for image labeling.* IEEE Conference on Computer Vision and Pattern Recognition, 2004: p. 1-8.

27. Jin, Y. and S. Geman, *Context and hierarchy in a probabilistic image model.* IEEE Conference on Computer Vision and Pattern Recognition, 2006 : p. 2145-2152.

28. Shotton, J., Winn, J., Rother, C. and Criminisi, A., *TextonBoost: joint appearance, shape and context modeling for multi-class object recognition and segmentation.* European Conference on Computer Vision, 2006: p. 1-15.

29. Sudderth, E., Torralba, Antonio, Freeman, W. and Willsky, A., *Learning hierarchical models of scenes, objects, and parts.* International Conference on Computer Vision, 2005: p. 1-8.

30. Gupta, A. and L. Davis, *Beyond Nouns: exploiting prepositions and comparative adjectives for learning visual classifiers.* European Conference on Computer Vision, 2008: p. 1-14.

31. Sali, E. and S. Ullman, *Combining class-specific fragments for object classification.* British Machine Vision Conference, 1999: p. 879-882.

32. Kumar, M.P., P.H.S. Torr, and A. Zisserman, *Object cut.* Computer Vision and Pattern Recognition, 2005: p. 1-8.

33. Cao, L. and L. Fei-Fei., *Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes.* International Conference on Computer Vision, 2007: p. 1-8.

34. Berg, T. and D. Forsyth, *Animals on the web.* Computer Vision and Pattern Recognition, 2006: p. 1-8.

35. Russell, B., Efros, A., Slvic, J., Freeman, W. and Zisserman, A., *Using multiple segmentations to discover objects and their extent in image collections. Proc.* Computer Vision and Pattern Recognition, 2006: p 1-8.

36. Wang, X. and E. Grimson, *Spatial latent dirichlet allocation.* Advances in Neural Information Processing Systems, 2007: p. 1-8.

37. Li, L.-J., G. Wang, and L. Fei-Fei., *Optimol: automatic online picture collection via incremental model learning. Proc.* Computer Vision and Pattern Recognition, 2007: p. 1-8.

38. Fergus, R., Fei-Fei, L., Perona, P. and Zisserman, A., *Learning object categories from google image search.* International Conference on Computer Vision. 2005: p. 1816-1823.

39. Fergus, R., P. Perona, and A. Zisserman, *A visual category filter for Google images.* European Conference on Computer Vision, 2004: p242-256.

40. Hoiem, D., A. Efros, and M. Hebert., *Putting Objects in Perspective.* Computer Vision and Pattern Recognition, 2006: p. 1-8.

41. Rabinovich, A., Vedaldi, A., Galleguillos, C., Wiewiora, E. and Belongie, S., *Objects in context.* International Conference on Computer Vision, 2007: p. 1-8.

42. Li, L.-J. and L. Fei-Fei., *What, where and who? classifying events by scene and object recognition.* Proc. International Conference on Computer Vision, 2007: p. 1-8.

43. Delage, E., H. Lee, and A. Ng, *Automatic single-image 3D reconstructions of indoor manhattan world scenes.* Proc. International Symposium on Robotics Research, 2005: p. 305-321.

44. Delage, E., H. Lee, and A.Y. Ng, *A dynamic Bayesian network model for autonomous 3D reconstruction from a single indoor image. Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* 2006: p. 1-8.

45. Hoiem, D., A. Efros, and M. Hevert, *Automatic photo pop-up.* ACM Special Interest Group on Groupware 2005: p. 1-8.

46. Dowty, D.R., R.E. Wall, and S. Peters, *Introduction to Montague semantics*. 1981: Kluwer Academic Publishers.

47. Hasle, G., K.-A. Lie, and E. Quak, eds. *Geometric modeling, numerical simulation and optimization : applied mathematics at SINTEF*. Mathematics and Statistics. 2007, Springer: Berlin.

48. Barnard, K., Duygulu, P., Forsyth, D., Freitas, N., Blei, D. and Jordan, M., *Matching words and pictures.* The Journal of Machine Learning Research, 2003. **3**: p. 1107-1135.

49. Blei, D. and M. Jordan, *Modeling annotated data.* ACM Special Interest Group on Information Retrieval *03*, 2003: p. 1-8.

50. Carbonetto, P., N.d. Freitas, and K. Barnard., *A statistical model for general contextual object recognition.* European Conference on Computer Vision, 2004: p. 350-362.

51. Duygulu, P., Barnard, K., Freitas, N. and Forsyth, D., *Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary.* European Conference on Computer Vision, 2002: p. 97-112.

52. Martino, B.D., *An ontology matching approach to semantic web services discovery.* Lecture Notes in Artificial Intelligence, 2006. **4331**: p. 550-558.

53. Le, B.T. and R. Dieng-Kuntz, *A graph-based algorithm for alignment of OWL ontologies.* IEEE/WIC/ACM International Conference on Web Intelligence, 2007: p. 466-469.

54. Wolper, L.F., ed. *Physician Practice Management - Essential Operational and Financial Knowledge*. 2005, Jones and Bartlett Publishers.

55. Goh, C.H., *Context interchange : new features and formalisms for the intelligent integration of information.* ACM Transactions on Information Systems 1999. **17**(3): p. 270-293.

56. Wu, G., J.Z. Li, and L. Feng, *Identifying potentially important concepts and relations in an ontology.* Proceedings of the 7th International Semantic Web Conference, 2008: p. 33-49.

57. Deering, M. and H. Sowizal, *Java 3D API specification*. 1997: Addison-Wesley.

58. Chung, F.-l. and B.Y.M. Fung, *Fuzzy color quantization and its application to scene change detection.* Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval 2003: p. 157-162.

59. Ray, S. and R.H. Turi, *Determination of number of clusters in K-means clustering and application in color image segmentation.* Proceedings of the 4th international conference on advances in pattern recognition and digital techniques, 1999: p. 137-143.

60. Konstantinidis, K., A. Gasteratos, and I. Andreadis, *Image retrieval based on fuzzy color histogram processing.* Optics Communications, 2005. **248**: p. 375-386.

61. Yen, J. and R. Langari, *Fuzzy Logic: Intelligence, Control, and Information*. 1998: Prentice-Hall.

62. Casella, G. and R.L. Berger, *Statistical Inference - 2nd edition*. 2001: Duxbury Press.

63. Hartley, R. and A. Zisserman, *Multiple View Geometry in Computer Vision*. 2 ed. 2003: Cambridge.

64. Zhang, Z., Deriche, R., Faugeras, O. and Luong, Q., *A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry.* Artificial Intelligence Journal, 1994. **78**: p. 87-119.

65. Buckley, J.J. and L.J. Jowers, *Fuzzy dynamic programming*. Monte Carlo Methods in Fuzzy Optimization. 2008, Berlin/Heidelberg: Springer.

66. Kacprzyk, J. and A.O. Esogbue, *Fuzzy dynamic programming : main developments and applications.* Fuzzy Sets and Systems, 1996. **81**(1): p. 31-45.

67. Klaus, A., M. Sormann, and K. Karner, *Segment-Based Stereo Matching Using Belief Propagation and a Self-Adapting Dissimilarity Measure.* International Conference of Pattern Recognition, 2006. **3**: p. 15-18.

68. Fischler, M.A. and R.C. Bolles, *Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography.* Comm. Assoc. Comm. Mach., 1981. **24**(6): p. 381-395.

69. Dhome, M., ed. *Visual Perception Through Video Imagery*. Digital Signal and Image Processing Series. 2009, Wiley.

70. Kobler, J., U. Schoning, and J. Toran, *The graph isomorphism problem : Its structural complexity*. Progress in Theoretical Computer Science, ed. V. Ronald. 1993: Birkhauser.

71. Garey, M.G. and D.S. Johnson, *Computers and Intractability : a guide to the theory of NP-Completeness*. 1979, New York: Freeman & co.

72. Ullmann, J.R., *An algorithm for subgraph isomorphism.* Journal of Association for Computing Machinery, 1976. **23**(1): p. 31-42.

73. Cordella, L.P., Foggia, P., Sansone, C. and Vento, M., *A (sub)graph isomorphism algorithm for matching large graphs.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004. **26**(10): p. 1367-1372.

74. Battiti, R. and F. Mascia, *An algorithm portfolio for the sub-graph isomorphism problem.* Lecture Notes in Artificial Intelligence, 2007. **4638**: p. 106-120.

75. Foggia, P., C. Sansone, and M. Vento, *A performance comparison of five algorithms for graph isomorphism.* Proceedings of the 3rd IAPR TC-15 Workshop on Graph-based Representations in Pattern Recognition, 2001: p. 188-199.

76. Messmer, B.T. and H. Bunke, *Subgraph isomorphism in polynomial time.* Technical report TR-IAM, 1995. **95**(3): p. 1-33.

VITA

Hyun Soo Lee received his B.S. in industrial engineering from Sungkyunkwan University and his M.S. in industrial engineering from Pohang University of Science and Technology (POSTECH). He joined Texas A&M University in the fall of 2006 for graduate studies in industrial engineering, where he was associated with the Virtual Systems and Augmented Reality Laboratory (VARL).

Hyun Soo's research interests include virtual model ontology, methodology for manufacturing intelligence, simulation framework supporting manufacturing uncertainties and cooperative design framework. He submitted and published his journal papers in *IEEE Transactions on System, Man and Cybernetics*, *International Journal of Production Research*, *Computer-Aided Design*, *International Journal of Collaborative Enterprise*, *Virtual Reality* and *Journal of Manufacturing Systems*.

Hyun Soo may be reached at: Department of Industrial and System Engineering, 241 Zachary Engineering Research Center, Texas A&M University, 3131 TAMUS, College Station, TX. 77843-3131.